Performance-Oriented Logistics Assessment (POLA)

Users' Manual for the Logistics Decision Model (LDM), Version IV

James H. Bigelow

RAND | ARROYO CENTER

The research described in this report was sponsored by the United States Army, Contract No. MDA903-91-C-0006.

Library of Congress Cataloging in Publication Data

Bigelow, J. H.
Performance oriented logistics assessment (POLA). Users manual for the logistics decision model (LDM), version IV / James H. Bigelow ; prepared for the United States Army.
p. cm.
Includes bibliographical references (p.).
"R-3814-A."
ISBN 0-8330-1186-3
1. United States. Army—Equipment—Maintenance and repair—Computer simulation—Handbooks, manuals, etc. 2. Armed Forces—Equipment—Maintenance and repair—Computer simulation—Handbooks, manuals, etc. 3. North Atlantic Treaty Organization—Armed Forces—Equipment—Maintenance and repair—Computer simulation—Handbooks, manuals, etc. I. United States. Army. II. RAND Corporation. III. Title.
UC263.B53 1991b
335.8'028'8—dc20 91-32788
CIP

Published 1992 by RAND
1700 Main Street, P.O. Box 2138, Santa Monica, CA 90407-2138

R-3814-A

Performance-Oriented Logistics Assessment (POLA)

Users' Manual for the Logistics Decision Model (LDM), Version IV

James H. Bigelow

Prepared for the
United States Army

RAND

PREFACE

This document is one of four describing the Performance Oriented Logistics Assessment (POLA) project. The three companion documents are:

- *Performance-Oriented Logistics Assessment (POLA): Executive Summary*, R-3823-A, which provides an executive level overview of the POLA methodology;
- *Performance-Oriented Logistics Assessment (POLA): Preparing the Logistics Decision Model (LDM) for Use in Analyses*, N-3393-A, which explains how to calibrate LDM and how to build its input files; and
- *Performance-Oriented Logistics Assessment (POLA): Relating Logistics Functional Capacities to Resources and Costs*, N-3354-A, which discusses in some detail other parts of the methodology.

POLA was a project in the RAND Arroyo Center, sponsored by the DCSLOG's Directorate of Plans and Operations (DALO-PLA) to develop a prototype methodology that would help build the logistics portion of the Army five-year program. Our prototype methodology consists of the following elements: the Logistics Decision Model (LDM) described in this document, a general approach for estimating the capacities of CSS[1] units to perform logistics functions from their equipment on hand, and a model for estimating the costs associated with changing the equipment on hand in CSS units.

The Logistics Evaluation Agency (LEA) has adopted the elements of the prototype methodology, has built a "shell" to link it with existing Army data files (such as the Total Army Equipment Distribution Plan, or TAEDP), and is using the combined system on real Army logistics problems. The combined system is called Logistics Net Assessment (LNA). However, LNA is not currently a polished, user-friendly, fully supported system, nor does it deal with all the logistics resources it might. Support of LNA, and its further development is the responsibility of the Army.

This report will be of particular interest to users of the Logistics Decision Model. Users should not only have the LDM program and an IBM-

[1]Combat Service Support. These units perform logistics functions such as transportation, maintenance, and supply.

compatible PC to run it on, they should also have the input files for a LDM case (one may use the test case distributed with this report). Consulting only this report, a user should be able to run LDM and display and interpret the results. He should also be able to modify the input files to represent a limited range of changes in theater logistics, such as changes in repair times or capacities, or transportation times or capacities. If a user wishes to create cases that differ greatly from a case in hand (e.g., add or entirely reformulate a logistics function such as POL[2] distribution or maintenance), he should consult Ref. [1].

THE ARROYO CENTER

The Arroyo Center is the U.S. Army's federally funded research and development center (FFRDC) for studies and analysis operated by RAND. The Arroyo Center provides the Army with objective, independent analytic research on major policy and management concerns, emphasizing mid- and long-term problems. Its research is carried out in five programs: Policy and Strategy; Force Development and Employment; Readiness and Sustainability; Manpower, Training, and Performance; and Applied Technology.

Army Regulation 5-21 contains basic policy for the conduct of the Arroyo Center. The Army provides continuing guidance and oversight through the Arroyo Center Policy Committee (ACPC), which is co-chaired by the Vice Chief of Staff and by the Assistant Secretary for Research, Development, and Acquisition. Arroyo Center work is performed under contract MDA903-91-C-0006.

The Arroyo Center is housed in RAND's Army Research Division. RAND is a private, nonprofit institution that conducts analytic research on a wide range of public policy matters affecting the nation's security and welfare.

Lynn E. Davis is Vice President for the Army Research Division and Director of the Arroyo Center. Those interested in further information about the Arroyo Center should contact her office directly:

Lynn E. Davis
RAND
1700 Main Street
P.O. Box 2138
Santa Monica, CA 90407-2138

[2]Petroleum, oil, and lubricants.

SUMMARY

This report is a user's manual for the Logistics Decision Model (LDM). It is distributed with a stand-alone version of the LDM program and the input and output files for a test case.

THE PERFORMANCE ORIENTED LOGISTICS ASSESSMENT (POLA) PROJECT

LDM was developed by the Performance Oriented Logistics Assessment (POLA) project as part of a larger methodology [2]. This project, under the sponsorship of the Army DCSLOG's Directorate of Plans and Operations (DALO-PLA), has developed a prototype methodology to help build the logistics portion of the Army five-year program. The POLA methodology estimates the effects on combat performance of alternative investments in logistics resources. If an increment of one resource has little effect on combat performance, and an equal-cost increment of a second resource has a large effect, the Army may prefer to invest less in the first resource and more in the second.

LDM plays a central role in the POLA methodology. It simulates the ways that Red and Blue combat forces are influenced by CSS[1] capacities (e.g., transportation, ammunition handling, maintenance) and logistics resources (e.g., stocks of ammunition, war reserve equipment, replacement personnel). The user represents logistics modifications[2] as changes in capacities or available resources from a base case. By comparing a logistics improvement case with the base case, he can estimate the effects of the logistics changes on combat performance measures such as FLOT[3] movement, Red and Blue weapons engaged and attrited, and Red and Blue resources consumed and personnel lost. The user can also observe such indicators of logistics "health" as maintenance queues, vehicles abandoned for lack of recovery assets, and excess capacities.

By itself, LDM cannot do all that is required of the POLA methodology. It can estimate the effect on combat performance of varying the capacities to perform certain logistics functions, such as ammunition

[1]Combat Service Support.

[2]Modifications may be either improvements or reductions.

[3]Forward Line of Own Troops.

or POL handling, but those capacities must themselves be estimated from physical resources. For example, the ammunition handling capacity of a Direct Support (DS) Ordnance Company must be estimated from the numbers of its fork lifts and cranes, and their operators. In addition, one must estimate the dollar costs of these resources. These tasks are performed by other models in the POLA tool kit [3].

LOGISTICS NET ASSESSMENT

The methodology developed by the POLA project has been adopted by the U.S. Army Logistics Evaluation Agency (LEA). The Operations Research Systems Analysis (ORSA) Support Team at LEA is responsible for further development and implementation of methodology, for maintaining data files, and for periodically recalibrating LDM.[4] They have created a Logistics Net Assessment (LNA) system, which they provide to action officers in the Pentagon. The LNA system includes the LDM program and a set of LDM input files, plus an input processor, an output analyzer, and a graph generator. The parts of the LNA system are tied together with DOS batch files.

USING THE LOGISTICS DECISION MODEL

Some Statistics

LDM has been designed to operate on an IBM-compatible personal computer. It is fast enough (perhaps ten minutes per case, depending on the size of the problem and the speed of the PC) that many variations on a campaign can be simulated at a single sitting. It is written in RM/FORTRAN, Version 2.4. The compiled program occupies about 380 kilobytes of memory and hence will run comfortably on a PC with 512K memory. The PC should have either two floppy disk drives or one floppy and one hard disk.

Three Steps in Using LDM

Using LDM in the form distributed with this report involves three steps. In the first step, the user sets up the input files with a

[4]To date, LDM has been calibrated to cases from CEM (Concepts Evaluation Model) and FORCEM (Force Evaluation Model). The Concepts Analysis Agency (CAA) regularly uses these large theater simulation models.

commercial text editor. In the second step, the user runs his simulation using the LDM program, and the outputs are automatically saved on disk. In the third step, the user pulls the outputs into a commercial spreadsheet/graphics package for further processing and display as desired.

Using a commercial text editor (to set up and modify the input data sets) and a commercial spreadsheet/graphics package (to display the results) eliminates the need to write new code for these functions, but it does lead to the inconvenience of switching between the different programs and of keeping track of the data files they generate. The user bears the entire burden of maintaining an audit trail from his final displays back to the original inputs.

As an alternative to using LDM in the form distributed with this report, a user may request the LNA system from the ORSA Support Team at LEA. If the user has the LNA system instead of the stand-alone version of LDM distributed with this report, this user's manual will still be useful, but the LNA System User's Guide [4] will still be needed as well.

LDM Input Files

LDM reads data from four different types of input files. The *ATTRITION* file contains data that relate the outcomes of combat (e.g., FLOT movement, attrition, consumption of resources, casualties) to the numbers and kinds of weapons engaged. The *SUPPORT* files (there may be more than one) describe the structure and activities that support the combat forces. These include repair of equipment; medical treatment of wounded personnel; and transportation of people, equipment, ammunition, and other resources. The *TIME_PHASE* file specifies the amount of each resource that enters the simulation at each time during the simulation. The *OUT_SPEC* files specify what variables appear in the output files. LDM allows the user to specify up to ten OUT_SPEC files, each of which can call for as many as 220 output variables.

How LDM Works

LDM contains two modules that perform computations. The *support* module computes the rates of all activities performed by the theater support structure to make weapons available for combat. Examples are the repair of tanks at Division Support Command (DISCOM) and

the transport of artillery ammunition from THEATER to Corps Support Command (COSCOM). The *combat* module computes the outcomes of combat between the available Blue and Red weapons. Examples of combat outcomes are FLOT movement, tanks damaged in combat, and artillery ammunition consumed.

Activities (and many combat outcomes) are defined by their effects on resources, such as consumption, production, or transport. Resources may be physical stocks (e.g., artillery ammunition), or they may represent capacities to perform activities (e.g., a repair capacity) or limits on stocks that one echelon can requisition from another. LDM constrains the rates of activities so that they will not consume more of any resource than is available.

To simulate the time required to perform an activity (e.g., a transportation time), LDM treats the activity as a pipeline. If a resource is consumed by an activity, all that is consumed in a time period will be subtracted from the available amount in that time period. But if a resource is produced by the activity, only a fraction of the amount produced in a time period is made available in that time period. The rest is made available in subsequent time periods. The fraction withheld is adjusted so that it takes the desired average time for the resource to pass through the pipeline.[5]

The activities have priority numbers, which determine the order in which LDM calculates their rates. At the start of each simulated time period, LDM adds resources scheduled to enter the simulation during that period and calculates the resources available for activities to use. Then the support module calculates the rates of activities with negative priority numbers. In the test case, these activities combine available resources at the BRIGADE echelon into weapons available for combat. For example, a tank cannot enter combat unless a crew is available, plus specified amounts of tank ammunition and other ammunition. Shortages of resources thus reduce the number of weapons available for combat.

Next, the combat module estimates the outcomes of one 12-hour period of combat between the available Blue and Red weapons. Outcomes that consume resources, such as consumption of ammunition and loss of or damage to tanks, are represented by activities with priority numbers of zero.

[5]For mathematical simplicity, I have assumed that the time to traverse the pipeline has an exponential distribution.

Finally, LDM's support module, invoked for a second time, calculates the rates of the activities with positive priority numbers. These include the recovery, evacuation, and repair of damaged equipment and the resupply of equipment, ammunition, and personnel.

Performing an Analysis with LDM

Using the POLA methodology, of which LDM is an important part, one can analyze and compare possible logistics modifications. An analysis consists of the following steps:

- Establish a base case that represents the current situation or some other desired reference situation;
- Define excursion cases to estimate the effects on combat performance measures of possible logistics modifications to the base case;
- Design LDM cases to investigate the sensitivity of conclusions drawn from the excursion cases to changes in uncertain parameters and assumptions; and
- Estimate the costs associated with each logistics modification, using other components of the POLA (or Logistics Net Assessment) methodology.

Establish a Base Case. It is important in any analysis to establish one case as a *base case* that serves as a point of departure and comparison for all other cases. For example, if one were analyzing logistics improvements to the U.S. Army in Europe as of 1989, then the capabilities of the weapons, the support structure, and the amounts of resources in the base case should all reflect those of the U.S. Army in Europe as of 1989.

I provide a test case with the LDM program distributed with this report, which may be used as a base case in practice analyses with LDM. The LNA system is distributed with its own base case, which is replaced each time LDM is recalibrated. The user may contact the ORSA Support Team at LEA for details.

Table S.1 shows the resources included in the test case. These resources may appear at any of four echelons: BRIGADE; DISCOM; COSCOM; and THEATER.[6] Resources at the BRIGADE echelon represent combat units available to fight. BRIGADE resources may

[6]If a user needs to represent a theater with a different organization, he can modify the LDM input files to have a different number of echelons.

Table S.1

Resources Considered in the Test Case

Supplies	Personnel	Equipment
	Combat Resources	
Tank ammo	Weapon crew	Tank
Artillery ammo	Combat personnel	APC
Other ammo	Helo crew	Helo
	Artillery personnel	ATM
	Driver	Artillery
		Recovery
		HET
	Support Resources	
	Driver	Truck
	Maintenance personnel	
	Medic	
	Supply personnel	
	Ordnance personnel	

be combined into eight kinds of weapons: (1) tanks, (2) Armored Personnel Carriers (APCs), (3) helicopters, (4) Antitank/Mortars (ATMs), (5) infantry, (6) artillery, (7) recovery vehicles, and (8) Heavy Equipment Transporters (HETs). Resources at the COSCOM and THEATER represent war reserves of combat equipment and ammunition, as well as support resources needed to perform the logistics support activities.

Figure S.1 shows some measures of combat performance from the test case. Except for the engaged tanks, all the measures in the figure are cumulated over time.

Define Excursion Cases. Typically, one establishes the base case to determine what logistics improvements will enhance combat performance, or what logistics degradations will have the least detrimental effect on combat performance. Each excursion case should be constructed by modifying the base case to reflect a possible logistics improvement or degradation. For example, one could add war reserve ammunition or more capacity to perform a given support task (e.g., ammunition handling). Other logistics improvements may involve changes to activity times (e.g., repair times) or to the amount of a resource consumed by an activity (e.g., maintenance manhours).

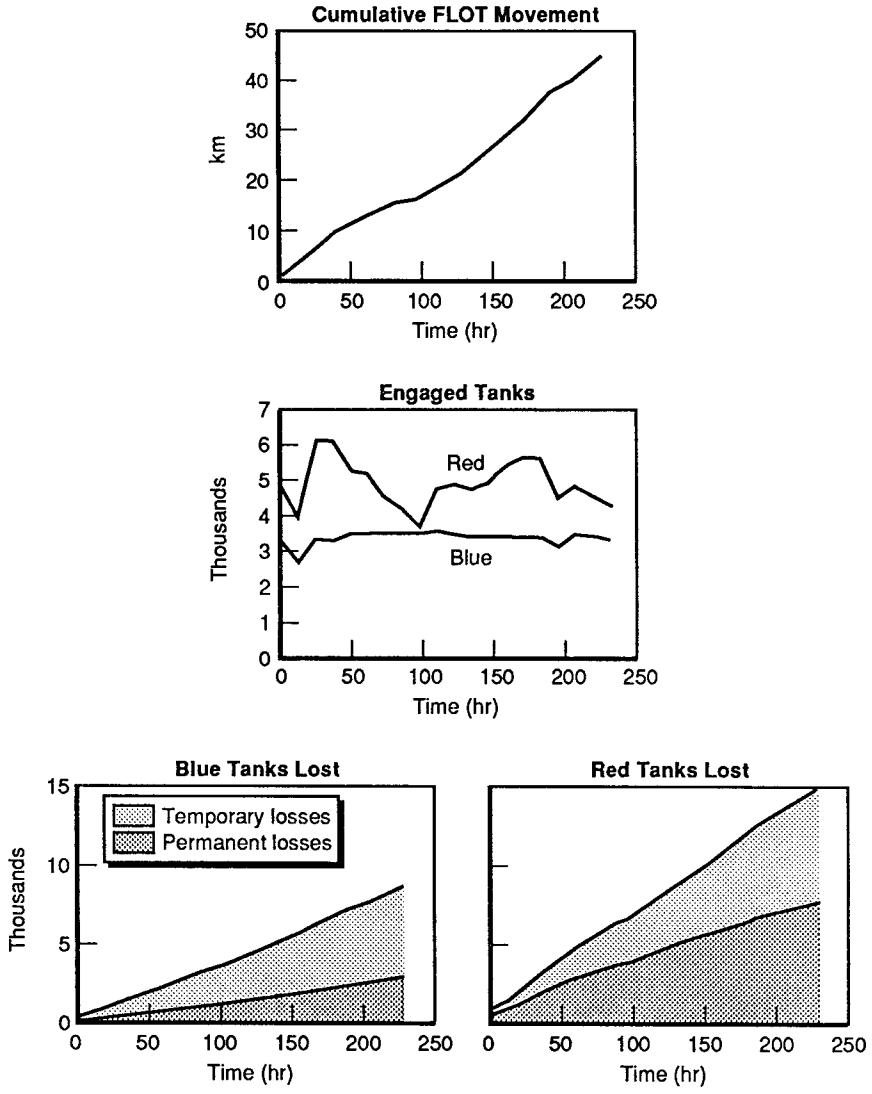

Fig. S.1—Some Combat Performance Measures

A preprocessor is often needed to translate a real-world logistics modification into terms the model can use. For example, suppose one is considering the addition of variable reach fork lifts to DS Ordnance companies, but fork lifts are not represented explicitly in the LDM

input files. However, there is a parameter that represents the capacity to handle ammunition at the DISCOM. The preprocessor would estimate the effect of adding fork lifts on the capacity parameter. A preliminary version of such a preprocessor is part of the LNA system [4].

Even without the preprocessor, a partial analysis can be carried out by directly varying the parameters in the LDM input files. Artillery ammunition at BRIGADE is depleted by the end of many of the simulated time periods in the test case. If more of this resource could be supplied to the BRIGADE, more ammunition could be fired and Blue's combat performance might improve.

With selected LDM outputs, the shortage of artillery ammunition at BRIGADE can be traced to a shortage of ammunition handling capacity at the DISCOM. I therefore construct a sequence of excursion cases by adding successively larger increments of ammunition handling capacity to the base case. I also extend this sequence in the opposite direction, decreasing the capacity from the base case. Table S.2 shows the number of divisions and the ammunition handling capacity per division over time in the excursion cases.

Figure S.2 shows some example results from these six excursion cases. They suggest that increases in ammunition handling capacity are of little value to Blue. However, decreases will cause him to lose ground to Red (cases E0, E1, and E2). At the same time, decreases in ammunition handling capacity cause Blue to withhold weapons from combat and therefore to suffer fewer losses of personnel and equipment. The user must decide whether loss of territory or the benefit of decreased personnel and equipment losses is more important.

Table S.2

Ammunition Handling Capacity per Division in the Excursion Cases

Time (hr)	Divisions	Handling Capacity (tons/division-day) in Case					
		E0	E1	E2	Base	E4	E5
0	10	0	720	1440	2160	2880	3600
24	12	0	700	1400	2100	2800	3500
36	13	0	692	1385	2077	2769	3462
204	14	0	686	1371	2057	2743	3429

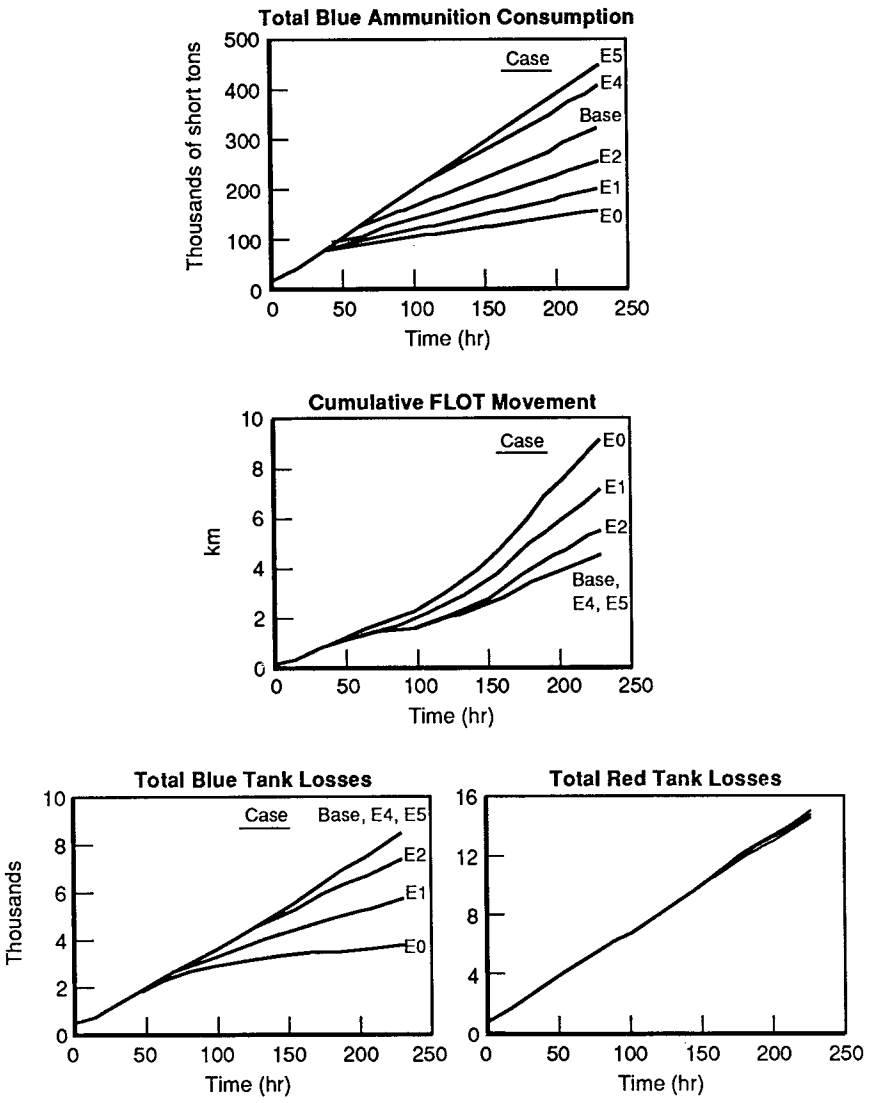

Fig. S.2—Results from Excursion Cases

Design Sensitivity Cases. Sensitivity cases are intended to test the effects of changes in uncertain parameters on the results estimated for logistics modifications. For example, there is a great deal of disagreement about the correct ammunition consumption rates, and

the user might wish to try several rates in different LDM cases. If a particular logistics modification is *robust*, it will make little difference how the uncertain parameters vary. That improvement will still yield substantial benefits. If the improvement is not robust, changing the uncertain parameters may eliminate the benefit.

Estimate the Cost of Each Logistics Improvement. LDM does not estimate costs, but cost estimation is a necessary part of any analysis. No matter how large a benefit a logistics improvement may yield, it may still turn out to be worse than an alternative improvement that is less costly. The POLA methodology currently includes a rudimentary cost model [3].

ACKNOWLEDGMENTS

The author is grateful to the following people for their contributions.

- The ORSA Support Team at the Logistics Evaluation Agency uncovered errors and suggested numerous improvements in early versions of the POLA methodology. They have incorporated the latest versions of the models in their Logistics Net Assessment system.
- Kenneth Girardini and John Bondanella of RAND reviewed this report and made many helpful suggestions for its improvement.
- Irene Gordon prepared the manuscript for publication.

CONTENTS

FIGURES

TABLES

GLOSSARY OF KEY WORDS

This glossary contains all the key words used in the LDM input files. The first listing is alphabetical and identifies which input files the words may appear in. The second listing is in order by type of input file and within each file alphabetically by key word. The second listing briefly describes what function the word performs.

Key Words and Input Files That May Contain Them

	CONTROL (Sec. 3)	ATTRITION (Sec. 4)	SUPPORT (Sec. 5)	OUT_SPEC (Sec. 6)	TIME_PHASE (Sec. 7)
A_INCR				X	
ACTIVITY				X	
ATTRITION	X	X			
AVAIL				X	
BLUE				X	
CATEGORIES		X			
CBT_LOSS		X			
COEF			X		
COMBAT				X	
CONSTR			X		
DEP		X			
END	X	X	X	X	X
EXOG				X	
FREQ_OUT	X			X	
FROM			X		
INHERIT				X	
INITIAL				X	
MINIMUM				X	
MOVEMENT		X			
OUT_SPEC	X			X	
PIPE			X		
POSTURE		X			
R_INCR				X	
RED				X	
REMAIN				X	
RHS			X		
STOCK			X		
SUPPORT	X		X		
TIME					X
TIME_PHASE	X				X
TO			X		
UNUSED				X	

Functions of Key Words

CONTROL FILE (Sec. 3)	
ATTRITION	Identify name of an ATTRITION file.
END	Signal end of CONTROL file.
FREQ_OUT	Specify simulated hours between outputs.
OUT_SPEC	Identify name of an OUT_SPEC file.
SUPPORT	Identify name of a SUPPORT file.
TIME_PHASE	Identify name of a TIME_PHASE file.
ATTRITION FILE (Sec. 4)	
ATTRITION	Signal start of data for calculating weapon systems hit on each side in each combat posture.
CATEGORIES	Signal start of data defining combat worth categories.
CBT_LOSS	Signal start of data defining how combat losses of resources will be calculated.
DEP	Used with key word CBT_LOSS to identify a resource whose losses in combat will be calculated.
END	Used with key words ATTRITION and CBT_LOSS to signal end of a section of input data.
MOVEMENT	Signal start of data defining FLOT movement by combat worth category and combat posture.
POSTURE	Signal start of data defining combat posture for each combat worth category.
SUPPORT FILES (Sec. 5)	
COEF	Used with key word PIPE to identify a resource consumed by an activity (interchangeable with key word FROM).
CONSTR	Define a "constraint" resource.
END	Signal end of a support file.
FROM	Used with key word PIPE to identify a resource consumed by an activity (interchangeable with key word COEF).
PIPE	Define a support activity.
RHS	Used with key words STOCK and CONSTR to define how to calculate the amount of the resource available.
STOCK	Define a "stock" resource.
SUPPORT	Signal start of data describing theater support structure.
TO	Used with key word PIPE to identify a resource produced by an activity.
OUT_SPEC FILES (Sec. 6)	
A_INCR	Request output of the net amount of a resource consumed by activities during an output cycle.
ACTIVITY	Request output of the sum of the rates of an activity in all simulation time periods in an output cycle.
AVAIL	Request output of the amount of a resource available to be consumed during the output cycle.

Functions of Key Words—continued

BLUE	Identify side for which an output quantity is requested.
COMBAT	Request output of variables calculated in the combat module of LDM, namely, weapons available, weapons engaged, and weapons hit by enemy fire.
PEND	Signal end of an OUT_SPEC file.
PEXOG	Request output of the amount of a resource supplied from outside the simulation (the TIME_PHASE file).
REQ_OUT	Specify simulated hours between outputs (applies to this OUT_SPEC file only).
INHERIT	Request output of the amount of a resource carried over from previous output cycle.
INITIAL	Request output of the amount of a resource available at the start of an output cycle.
MINIMUM	Request output of the smallest amount of a resource remaining at the end of any simulation time period in an output cycle.
OUT_SPEC	Signal start of data in OUT_SPEC file.
R_INCR	Request output of the amount of a resource generated by resource matrix calculations.
PRED	Identify side for which an output quantity is requested.
PREMAIN	Request output of the amount of a resource remaining at the end of the output cycle.
PUNUSED	Request output of the sum of the amounts of a resource that remained at the ends of all simulation time periods in an output cycle.
TIME_PHASE (Sec. 7)	
END	Signal end of TIME_PHASE file.
TIME	Signal start of data for next simulated time period.
TIME_PHASE	Signal start of data in the TIME_PHASE file.

1. INTRODUCTION

This report is a user's manual for the Logistics Decision Model (LDM). The model was developed by the Performance Oriented Logistics Assessment (POLA) project as part of a larger methodology. The overall POLA project is described in a companion publication [2], but to provide context for LDM we summarize it here.

THE POLA PROJECT

Performance Oriented Logistics Assessment, a project sponsored by the Army DCSLOG's Directorate of Plans and Operations (DALO-PLA), has developed a prototype methodology to help build the logistics portion of the Army five-year program. When building its program, the Army first estimates a requirement for each resource, but the price of satisfying all requirements always greatly exceeds the amount the Army can spend. The Army then prioritizes the requirements, deciding which will be funded immediately and which will be deferred, perhaps for several years. The Army has always made these decisions on somewhat arbitrary grounds, for it has never succeeded in developing tools that would systematically rate different resources, intended to support disparate functions, on common scales.

The POLA methodology attempts to rectify this lack by estimating effects on combat performance of alternative investments in logistics resources. Combat performance measures thus become the common scales on which different resources are rated. If an increment of one resource has little effect on combat performance, and an equal-cost increment of a second resource has a large effect, the Army may prefer to satisfy less of the requirement for the first resource and more of the requirement for the second. Combat performance is measured in terms of FLOT movement, Red and Blue weapons engaged and attrited, and Red and Blue resources consumed and personnel lost. Logistics resources considered include stocks of ammunition, POL, war reserve equipment, and replacement personnel. Also considered are resources that increase CSS capacities, such as capacities to handle ammunition, transport dry cargo, etc.

THE POLA METHODOLOGY

The POLA methodology consists of several small models, as shown in Fig. 1. The LDM is intended to estimate the effects of logistics improvements on combat performance. It simulates the ways that Red and Blue combat forces are influenced by CSS capacities (e.g., transportation, ammunition handling, maintenance) and logistics resources (e.g., stocks of ammunition, War Reserve Materiel (WRM) equipment, replacement personnel).

Figure 2 shows LDM's inputs and outputs. The user represents logistics modifications as changes in capacities or available resources from a base case. By comparing a logistics modification case with the base case, he can estimate the effects of the logistics changes on combat performance measures. The user can also observe indicators of logistics "health," such as maintenance queues, vehicles abandoned for lack of recovery assets, excess capacities, etc.

By itself, LDM cannot do all that is required of the POLA methodology. It can estimate the effect on combat performance of varying the capacities to perform certain logistics functions, such as ammunition

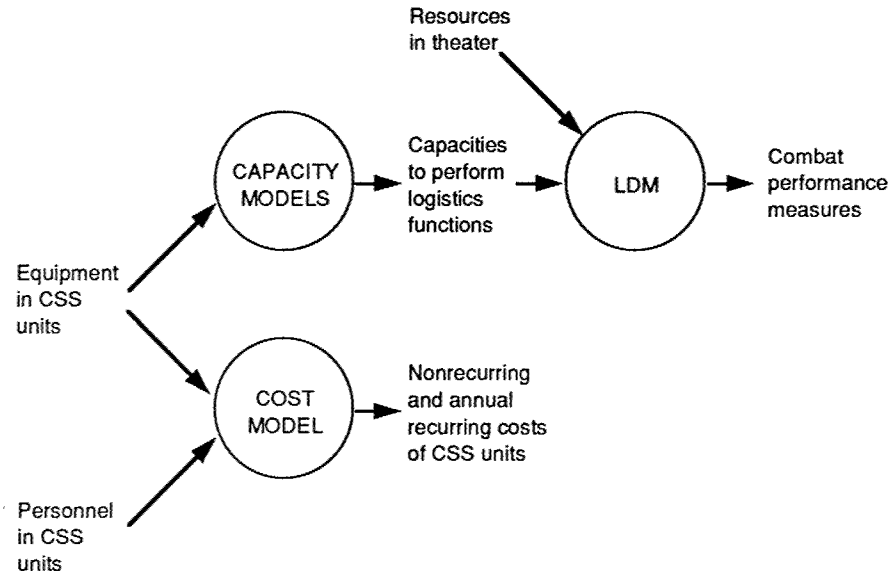

Fig. 1—Overview of the POLA Methodology

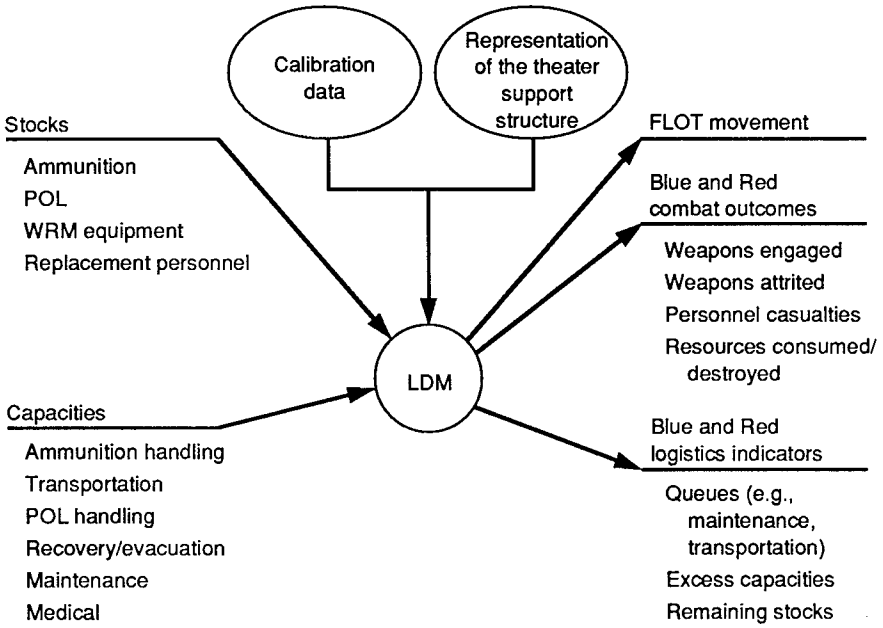

Fig. 2—LDM Inputs and Outputs

or POL handling, but those capacities must themselves be estimated from physical resources. For example, the handling capacity of a DS Ordnance Company must be estimated from the numbers of its fork lifts and cranes, and their operators. In the POLA methodology, this is done by capacity models (see Fig. 1).

One must further estimate the dollar costs of these resources. The cost estimating procedures must estimate much more than the current purchase price of a particular resource. Buying a resource (e.g., a variable reach fork lift) commits the Army to other expenditures as well (e.g., for training of crews and maintenance personnel, fuel, storage facilities). The procedures can estimate the overall cost of equipping and fielding a new unit, or they can be applied to incremental resources to estimate the cost of adding resources or people to an existing unit. Cost estimation is performed by a cost model (Fig. 1). Both the cost model and the capacity models are described in Ref. [3].

It is also necessary to build a base case and several logistics improvement cases for LDM. The base case should reflect the Army programmed for the analysis year (the current year or a specified future year). Logistics improvement cases are best defined as modest changes from the units and resources the Army possesses in the base case. Reference [3] identifies Army databases that describe the current Army and the Army that is programmed for the future, and procedures are worked out for drawing from them the necessary unit and resource data. Most important among these databases are:

- TAEDP (Total Army Equipment Distribution Program);
- AMDF (Army Master Data File);
- TOEs (Tables of Organization and Equipment); and
- FAS (Force Accounting System).

Some of these files are classified, so analyses using the POLA methodology are generally also classified.

LOGISTICS NET ASSESSMENT

The methodology developed by the POLA project has been adopted by the U.S. Army Logistics Evaluation Agency. The ORSA Support Team at LEA is responsible for further development and implementation of methodology, for maintaining data files, and for periodically recalibrating LDM and providing it to action officers in the Pentagon. They have created an LNA system that consists of:

- An input processor, written as a dBase III application;
- The LDM program;
- An output analyzer in the form of Lotus 1-2-3 spreadsheets with macros; and
- A graph generator, which uses Lotus Graph Writer.

These modules are integrated through the use of DOS batch files. Users interested in obtaining the entire LNA system should contact the ORSA Support Team at LEA. If you have the LNA system instead of the stand-alone version of LDM distributed with this report, this user's manual will still be useful, but you should have the LNA System User's Guide [4] as well.

OPERATION AND USE OF LDM

LDM has been designed to operate on an IBM-compatible personal computer. It is fast enough (perhaps ten minutes per case, depending on the size of the problem and the speed of the PC) that many variations on a campaign can be simulated at a single sitting. It is written in RM/FORTRAN, Version 2.4. The compiled program occupies about 380 kilobytes of memory and will run comfortably on a PC with 512K memory. The PC should have either two floppy disk drives or one floppy and one hard disk.

There are several LDM input files for each case, all of which are ASCII files that can be operated on with most standard text editors or database systems. LDM does not include an editor, so the user must provide his own means to prepare input files. Any commercial text editor will serve if it can produce ASCII files with no special control characters. Similarly, LDM writes its outputs to ASCII files but provides no built-in routines for examining them or producing reports. Rather, the user should import the output files into a commercial spreadsheet/graphics package. There he can manipulate the outputs, create graphs, even combine outputs from several cases.

The support structure simulated by LDM is almost entirely specified by the inputs. Very little is hard-wired into the program itself. LDM assumes that the support structure can be represented as a network. Links in the network represent activities that move resources from one location to another (e.g., transportation of ammunition) or transform resources from one condition to another (e.g., repair of equipment). Nodes represent the resources at various locations and in various conditions that are moved or transformed by the activities. Capacities can be imposed on links or groups of links. For example, the repair of each type of equipment at the DISCOM is a different activity, but a single capacity constraint may be imposed on all of them.

The test case distributed with LDM illustrates how the LDM inputs are used to define the problem to be simulated. It also provides a structure that the user may modify to create his own simulations. However, if the user wishes to define simulations that differ greatly from the test case, he should consult Ref. [1].

Performing an analysis with LDM requires much more than defining a single simulation, running it, and displaying the results. An analysis should include many simulation cases, including a base case to represent the current situation (or some other reference point) and

excursion cases that incorporate incremental logistics changes from the base case. Results from excursion cases should be compared with base case results to determine the effects of the logistics changes on combat performance. One must devise a way to generate interesting new excursion cases and establish a scheme for keeping track of all the cases generated.

ORGANIZATION

Section 2 describes how to operate LDM. Sections 3 through 7 describe the various LDM input files, including formats for all types of records that may occur in them. Section 8 discusses the test case distributed with LDM. Section 9 discusses how to carry out an analysis of possible logistics improvements with LDM.

2. OPERATION OF LDM

REQUIREMENTS FOR USING LDM

The LDM program will run on an IBM PC or compatible machine with at least 512 kilobytes of RAM. It is desirable for the machine to have a hard disk, but LDM can be run on a machine with two floppy disk drives. The user will also need a text editor or word processor to prepare and modify input files. Any text editor (or other software) will serve if it can read, modify, and write sufficiently large ASCII files. Finally, the user will need software with which to view and manipulate the output files. I use Lotus 1-2-3 for the last purpose, although other spreadsheet programs ought to be equally effective.

Figure 3 shows how the text editor, the LDM model, and Lotus 1-2-3 are used together to perform an analysis. Using the text editor, the analyst creates new input files for LDM or revises old ones. The

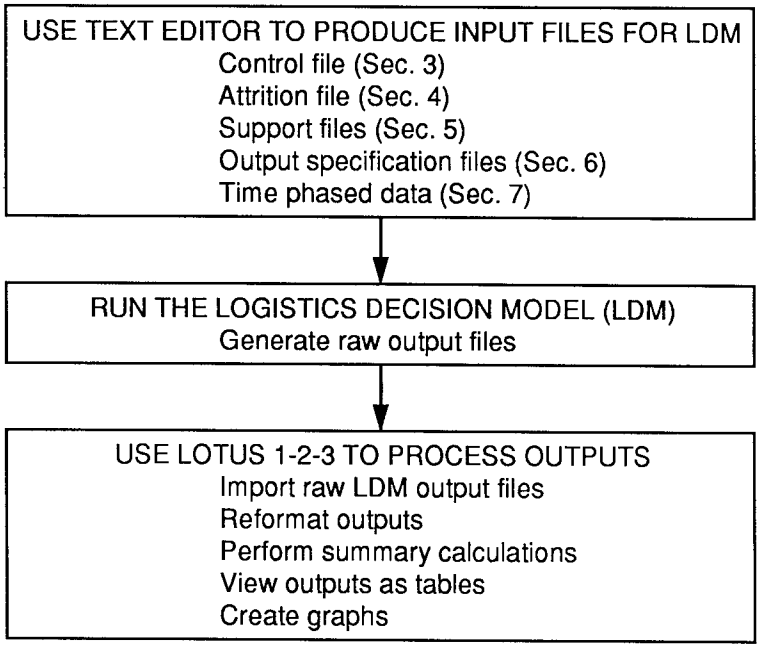

Fig. 3—Using LDM for an Analysis

different types of LDM input files are named in the figure, along with the sections of this report that describe them. The user then runs the LDM program, producing raw output files. Finally, he imports the raw output files into Lotus 1-2-3 to reformat and summarize them and to view the results as tables or graphs.

Using a text editor (to set up and modify the input datasets) and Lotus (to display the results) eliminates the need to write new code for these functions, but it does lead to some user inconvenience. When the user wishes to change inputs or to examine the results of an LDM simulation, he must load the text editor or Lotus into his PC, overwriting LDM. Upon examining the results of one simulation, the user will often wish to generate another. To do so, he must load and use the text editor, the LDM program, and Lotus. This will entail a delay of several minutes; and if the user's PC has no hard disk, it will also entail some swapping of disks.

In addition, relying on existing software packages exacerbates a problem of proliferating files. LDM requires four kinds of input files (see Sec. 3) and generates two kinds of output files.[1] It is hard enough to keep track of what LDM input files are associated with which output files. But on top of this, the use of Lotus will generate worksheet and graph datasets as well. Moreover, on many occasions, the user will probably construct Lotus worksheets or graphs that contain data from two or more LDM simulations. The user bears the entire burden of maintaining an audit trail from his final displays back to the original inputs.

THE LNA SYSTEM

The ORSA Support Team at the Logistics Evaluation Agency (LEA) has built a system that automates much of the switching back and forth between software packages. Their system, called Logistics Net Assessment, consists of:

- An input processor, written as a dBase III application;
- The LDM program;
- An output analyzer, in the form of Lotus 1-2-3 spreadsheets with macros; and

[1]One is a log that records the names of input files used for a run, as well as any error messages generated. The other kind of output file contains the actual results of the run. A single run of LDM can generate as many as ten of the latter kind of output file.

- A graph generator that uses Lotus Graph Writer.

These modules are integrated through the use of DOS batch files. Users interested in obtaining the entire LNA system should contact the ORSA Support Team at LEA.

If you have the LNA system instead of the stand-alone version of LDM distributed with this report, much of the material in this section will not apply to you. Refer to the LNA System User's Guide [4].

FILES DISTRIBUTED WITH LDM

The LDM program and samples of files needed to run it are included with this report on three 5.25 inch floppy disks, each formatted at a density of 360 kilobytes. Their contents are as follows:

Disk 1: PROGRAM

LDM.FOR	FORTRAN source code for LDM
LDM.INC	"Include" file for source code, containing common blocks
LDMCO.EXE	Version of executable LDM program that requires math coprocessor
LDMEM.EXE	Version of executable LDM program that does not need math coprocessor

Disk 2: SAMPLE OUTPUT

CBTLIM.SAM	First sample output file from test problem
BDERES.SAM	Second sample output file from test problem
LOSSES.SAM	Third sample output file from test problem
AMMO.SAM	Fourth sample output file from test problem

Disk 3: DATA FOR TEST PROBLEM

MASTER	Master control file for running test problem
ATT.DAT	Attrition input data for test problem
BATTLE.DAT	First support activity input file for test problem
MAINT.DAT	Second support activity input file for test problem
PER_SUP.DAT	Third support activity input file for test problem
NOMTIME.DAT	Input data for test problem describing time-phased entry of resources into the theater

CBTLIM.OUT	Defines the output file CBTLIM.PRN
BDERES.OUT	Defines the output file BDERES.PRN
LOSSES.OUT	Defines the output file LOSSES.PRN
AMMO.OUT	Defines the output file AMMO.PRN
MACRO.WK1	Lotus macro for converting .PRN output files into spreadsheet form (works with Lotus Version 2.01)

LDM can run on a PC with two floppy disk drives (drives A and B), or one floppy (drive A) and a hard disk (drive C). If you plan to use a two-floppy system, start by making backups of the three LDM disks.[2] If you plan to use a system with a hard disk, I recommend creating a new directory for LDM and copying these disks into it. You might call the directory C:\LDM.

LDM is distributed with the FORTRAN source code and two executable modules (.EXE files). One executable module, LDMCO.EXE, will work only on a PC that has a math coprocessor. It will produce an error message if one attempts to run it on a machine without a coprocessor. The other executable module, LDMEM.EXE, will run with or without a coprocessor.

Also in the distribution package are the input and output files for a test case. To make sure that your copy of LDM is working properly, run it as described below with the input files on the Data disk. It will produce four output files, named CBTLIM.PRN, BDERES.PRN, LOSSES.PRN, and AMMO.PRN. If you compare these with the CBTLIM.SAM, BDERES.SAM, LOSSES.SAM, and AMMO.SAM files on the Sample Output disk using the DOS COMP command, they should be identical.

RUNNING THE LDM PROGRAM

In the form distributed with this report, LDM can be run on a system with a hard disk and one floppy disk, or on a system with two floppy disk drives. Systems with a hard disk are more convenient, not only because data can be read from the hard disk faster than from a floppy disk, but because the hard disk holds much more data. If you do much analysis with LDM, you will run many cases and generate many output files. To do this on a two-floppy system requires a large

[2]This is the usual good advice that most of us ignore from time to time, with occasionally disastrous consequences.

supply of diskettes and much switching of diskettes in the floppy disk drive.

Running LDM from a Hard Disk

If your system has a hard disk, copy the three distribution diskettes onto it, as mentioned above. Make sure that the input files, the MASTER file, and the LDM program are in the same directory on drive C, and change to that directory. Then type:

LDMCO

(If your system does not have a coprocessor, type LDMEM instead.) When the program asks:

Enter name of master control file

the user should answer:

MASTER

The control file MASTER can be found on Disk 3, which contains the data for the test problem. This file identifies the input and output files LDM is to access during its simulation run. Section 3 discusses the control file in detail.

Running LDM on a Two-Floppy System

To run LDM on a two-floppy system, place the Program disk (or a backup copy of it) in drive A, and the Data disk (or a copy of it) in drive B. Make sure that the PC is reading drive B (i.e., is prompting the user with "B>"), and type:

A:LDMCO

(If your system has no math coprocessor, type A:LDMEM instead.) The program will load, and almost immediately this message will appear:

Enter name of master control file

The user should then type:

MASTER

Possible Errors

One of two errors might occur at this point. First, there might not be a file named MASTER (or whatever was typed as the name of the master control file) in the directory from which LDM is being run. In that case LDM will issue error messages to the screen and to the log file REPROL.LOG.[3] The messages are:

Log : ERROR #(errno) ON OPENING CONTROL FILE filename)

Screen: Execution terminated: ERROR OPENING CONTROL FILE

In the message to the log file, (errno) is an error number assigned by the RM/FORTRAN compiler, and (filename) is the name the user typed for the master control file.

The other error will occur if the user names his control file REPROL.LOG. In this case the messages are:

Log : File names conflict. Rename master control file.

Screen: Execution terminated: FILE NAME CONFLICT

VIEWING THE OUTPUT WITH LOTUS 1-2-3

If LDM runs successfully, it will generate one or more output files as specified in the control file MASTER. As discussed earlier, the LDM model does not have the capability to display or manipulate outputs. I recommend using Lotus 1-2-3 to view LDM outputs, for several reasons.[4] First, Lotus can perform calculations on the outputs, such as cumulating losses over time or calculating hits as percentages of engaged weapons. Second, Lotus can display the output in graphical as well as tabular form. Third, Lotus can combine results from several LDM runs so that the user can compare them. It would be

[3]REPROL.LOG is a file automatically created by LDM each time it runs. If a file named REPROL.LOG already exists in the directory from which LDM is invoked, it will be overwritten, so if the user wishes to save the log file from an earlier run, he should rename it before running LDM again. LDM writes various messages to this file that describe the progress of the run. If an error for which I made explicit provision is found in one of the inputs, a message will be written to this file to help the user locate and correct it. Error messages generated by the system routines provided with the RM/FORTRAN compiler will appear only on the screen.

[4]Another spreadsheet package would serve as well. However, the keystroke sequences mentioned in this text will work only with Lotus 1-2-3. If you use a different spreadsheet package, you must substitute the appropriate keystrokes.

very costly to develop these capabilities anew as part of the LDM program.

Tell Lotus Where the Output File Is

Once you have loaded Lotus, you must tell it how to find the LDM output files. If you are using floppy disks, the Lotus disk may be in drive A while the LDM output file is on the disk in drive B. If you have a hard disk, both your Lotus software and the LDM program and data files may be there. However, they are probably in different directories—e.g., Lotus in a directory called C:\123 or C:\Lotus and the LDM files in a directory called C:\LDM. To tell Lotus where the LDM output files are, hit the key sequence /FD and then type the name of the drive and directory containing the LDM output. (In the above key sequence, the "slash" brings up the main Lotus menu, F selects the File submenu, and D selects the Directory command within that submenu.) If the LDM output is in drive B, type B:. If it is in directory C:\LDM, then that is what you type.

Load the MACRO.WK1 Worksheet

I have included a file called MACRO.WK1 on Disk 3, "Data for the Test Problem." This Lotus worksheet consists of a macro that reformats LDM output files (the files with the .PRN extenders), as described below.[5] Retrieve the MACRO.WK1 worksheet by hitting the key sequence /FR (for File Retrieve) and then either typing the name MACRO or highlighting MACRO.WK1 with the cursor. When Lotus retrieves the worksheet, the cursor will be at cell A17. Leave it there!

Import the LDM Output File

To tell Lotus you want to import your LDM output file, hit the key sequence /FIN, for "File Import Numbers." The "File Import" command (/FI of the above key sequence) is Lotus's way of reading an ASCII file, which is the format of the output file created by LDM. The final keystroke, N, tells Lotus to import numbers rather than text. To do this, Lotus needs alphanumeric fields in the file to be set off with quotes, and all fields to be separated by blanks or commas. LDM writes its output files to satisfy these requirements.

[5]This macro was written for Lotus 1-2-3 Version 2.01. It will not work with Lotus 1-2-3 Version 3. I have not tested it with Version 2.2.

At this point, Lotus will display the name of every file in your directory (selected with the /FD key sequence, as described above) that has a .PRN extender. These names will be displayed on a line near the top of the screen. If the directory contains too many of these names to show them all simultaneously, you can use the left and right arrow keys to scroll to the names that are initially off the screen. Use the arrow keys to place the cursor over the name of the file you wish to view, and press "Enter" ("Return" on some PCs). Lotus will display a "WAIT" message in the upper right corner of the screen while it imports your file. When the message changes to "READY", the file has been loaded. It is now a Lotus worksheet.

Reformat the Worksheet

Figure 4 illustrates the format in which LDM writes its output files. A block of lines at the top contains titles, followed by lines with labels of the output quantities (as specified in a .OUT file; see Sec. 3). Below this, blocks of data contain the values of these quantities, enough for all the simulated time periods from which outputs were collected, at ten time periods per block. (The last block may contain fewer than ten periods.) These blocks of data must now be brought up from below and arrayed to the right of the labels, as shown in Fig. 5. The Lotus macro in the MACRO.WK1 worksheet, which you have already loaded, will perform this task for you if you press <Alt-R> (hold down the Alt key and press R).

Once you have used the macro to reformat a worksheet, there is no point to keeping the macro instructions in that worksheet. You can get rid of them by moving the cursor to cell A1 (by hitting the "Home" key), hitting /WDR (Worksheet Delete Rows), moving the cursor down to the last macro instruction (cell A16), and hitting Enter. Lotus will then delete those rows.

Save the Worksheet

Now your LDM output data are in a Lotus spreadsheet. Before anything else, I strongly recommend that you save the worksheet. Use the key sequence /FS (for "File Save").

Caution! When you save the worksheet, Lotus will offer you the default name MACRO.WK1, because at the very beginning of the process you retrieved this file, establishing it as the name of the worksheet. Be sure you change the name, or you will overwrite

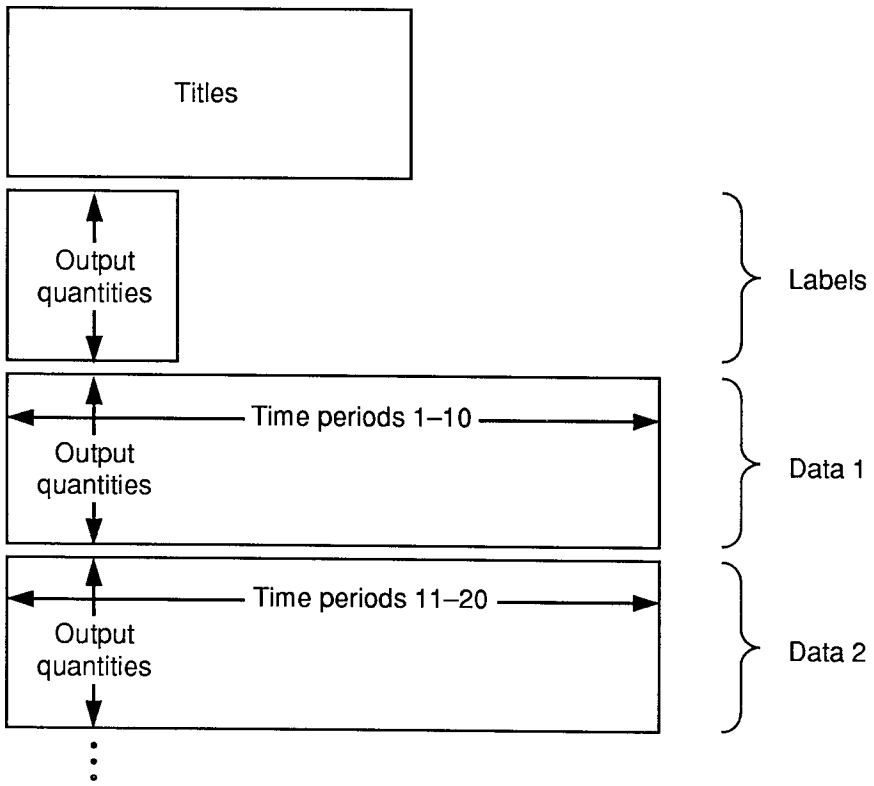

Fig. 4—Format of the LDM Output File

MACRO.WK1. A good name to choose is the name of the .PRN file from which the worksheet was built. For example, if you started with the output file CBTLIM.PRN, you could name the worksheet CBTLIM. Lotus will automatically supply the extender .WK1, so when you look for the worksheet on disk, it will have the name CBTLIM.WK1. Once you have completed these steps, you are ready to do some analysis.

TERMINATION CODES

It is possible to run LDM from a batch file. The user might wish to do this to automate the transition from the text editor that generates the LDM input files, to the LDM program itself, to the spreadsheet

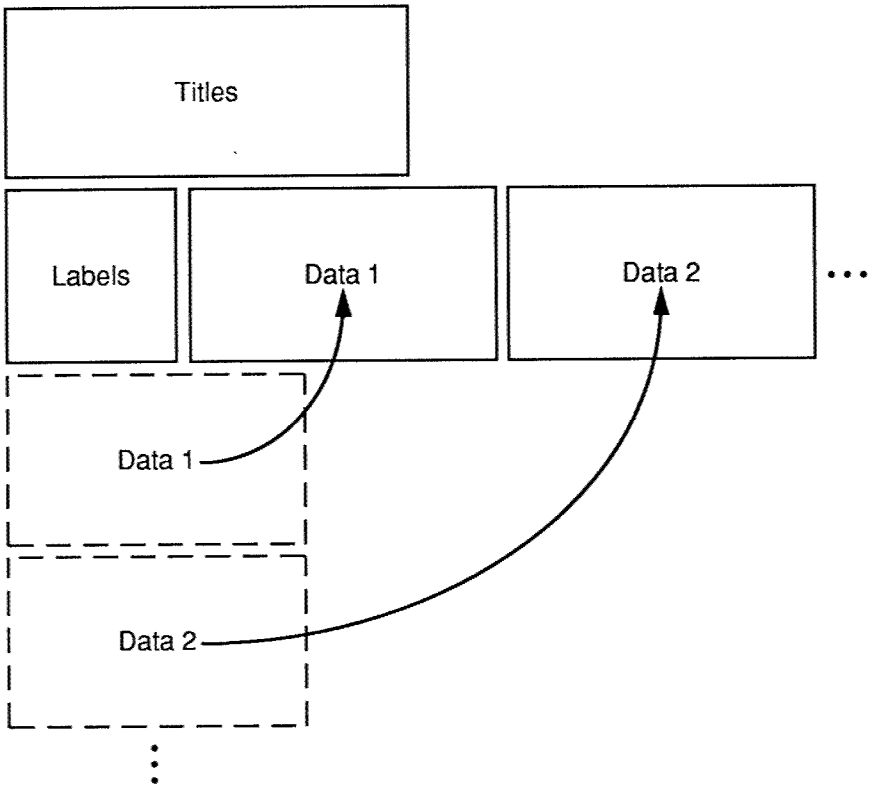

Fig. 5—Reformatting LDM Outputs in Lotus

program used to examine the outputs. However, one would not wish to transition from LDM to the spreadsheet program unless LDM has run successfully.

The DOS system variable called ERRORLEVEL can be tested to determine whether LDM has run successfully. If LDM terminates normally, ERRORLEVEL will equal zero. If it terminates with an error I have explicitly trapped in the LDM program, ERRORLEVEL will equal 1. If it terminates with an error I have not trapped (an error I have left for the RM/FORTRAN system routines to deal with), ERRORLEVEL will not be set by the LDM program. I have tried to trap all errors that might be encountered while reading the input files. An example of an error I have not trapped is the one that occurs when a user tries to run LDMCO on a PC with no math coprocessor.

The following sample batch file shows how ERRORLEVEL can be tested.

```
        ECHO OFF
        LDMCO
        IF ERRORLEVEL 2 GOTO TWO
        IF ERRORLEVEL 1 GOTO ONE
        IF ERRORLEVEL 0 GOTO ZERO
    :ZERO
        ECHO ERRORLEVEL = 0
        GOTO END
    :ONE
        ECHO ERRORLEVEL = 1
        GOTO END
    :TWO
        ECHO ERRORLEVEL >= 2.  NOT SET IN LDM
    :END
        ECHO ON
```

The user would substitute his own actions for the ECHO ERRORLEVEL = 0 and ECHO ERRORLEVEL = 1 statements. For example, he might replace ECHO ERRORLEVEL = 0 with a call to the spreadsheet program. Similarly, he might replace ECHO ERRORLEVEL = 1 with statements that list the REPROL.LOG file, which contains messages describing the kind of error that caused LDM to terminate, and in what input file it occurred.

3. THE CONTROL FILE

Overall control of an LDM run is exercised through a control file. For the test problem, the control file is called MASTER. Here is a listing of MASTER. The rules at the top and bottom of the listing indicate which columns are occupied by each data element.[1]

```
123456789_123456789_123456789_123456789_123456789_123456789_
------------------------------------------------------------
FREQ_OUT     12.
ATTRITION            ATT.DAT
SUPPORT              BATTLE.DAT
SUPPORT              MAINT.DAT
SUPPORT              PER_SUP.DAT
OUT_SPEC             CBTLIM.OUT          CBTLIM.PRN
OUT_SPEC             BDERES.OUT          BDERES.PRN
OUT_SPEC             LOSSES.OUT          LOSSES.PRN
OUT_SPEC             AMMO.OUT            AMMO.PRN
TIME_PHASE           NOMTIME.DAT
END
------------------------------------------------------------
123456789_123456789_123456789_123456789_123456789_123456789_
```

Each line in the control file has a key word in the first eight columns.[2] The above listing contains all the key words that LDM will recognize in the control file. LDM reads the control file one line at a time and processes that line before moving to the next. Processing involves writing the record to the screen (so the user knows something is happening) and to a file called REPROL.LOG and then reading and manipulating any associated data.[3]

---

[1]It is important to place each data element in its proper column. Generally, LDM will not recognize a misplaced datum.

[2]Any characters beyond the first eight are superfluous, such as the N in ATTRITION and the SE in TIME_PHASE.

[3]REPROL.LOG is a log file automatically created by LDM each time it runs. If a file named REPROL.LOG exists in the directory in which LDM is run, it will be overwritten, so if you want to save an old REPROL.LOG, rename it before running LDM. LDM will write each line of the control file to REPROL.LOG, as well as most error messages. If there are errors in the input files, for example, this information may help the user locate them.

THE ATTRITION, SUPPORT, AND TIME_PHASE COMMANDS

When LDM encounters any of the key words ATTRITION, SUPPORT, or TIME_PHASE, it will read data from the input file whose name starts in column 20 of that line. The file name must occupy 14 or fewer characters.

Later sections will discuss at length the input files named on the ATTRITION, SUPPORT, and TIME_PHASE command lines. Briefly, however, LDM consists of two major modules, a *combat* module and a *support* module. The combat module takes data from the input file named on the ATTRITION line (ATT.DAT in the above listing) plus Blue and Red inventories of weapons (calculated in the support module) and calculates FLOT movement; number of weapons that suffer hits; and amounts of various resources that are consumed, lost, or damaged. The support module takes these results from the combat module, plus:

- Data describing the support structure from the files named in the SUPPORT command lines (BATTLE.DAT, MAINT.DAT, and PER_SUP.DAT in the above listing); and
- Data describing the time-phased entry of resources into the theater from the file named on the TIME_PHASE line (NOMTIME.DAT in the above listing);

and estimates the amount of each support activity that can occur in a simulation time step. It also estimates the results of those activities, among which are the inventories of Blue and Red weapon systems available for combat in the next time period. This information is fed back to the combat module for the next simulation cycle.

THE OUT_SPEC COMMAND

During each cycle, various calculated quantities are collected for later output. When LDM encounters the key word OUT_SPEC, it reads data from the input file whose name starts in column 20 of that record and writes to the output file whose name starts in column 40. These file names must be no more than 14 characters long. The files named in column 20 of the OUT_SPEC command lines (CBTLIM.OUT, BDERES.OUT, LOSSES.OUT, and AMMO.OUT in the above listing) specify which output quantities are to be collected. A later section will discuss these files at length. The data called for in the input files will be written to the output files named in column 40 of those same lines

(CBTLIM.PRN, BDERES.PRN, LOSSES.PRN, and AMMO.PRN in the above listing).

THE FREQ_OUT COMMAND

When LDM encounters the FREQ_OUT key word, it sets the frequency with which output quantities will be collected during the simulation. The key word FREQ_OUT starts in column 1, and the output frequency (i.e., number of hours between successive output reports) should be in columns 13–18, and should include a decimal point. Legal output frequencies are 12 hours or any multiples of 12 hours (e.g., 24, 36, 48, etc.). If you enter a number smaller than 24 hours, LDM will assume a frequency of 12 hours. Any number larger than 24 hours will be rounded down to the next multiple of 12 hours (e.g., 28 hours rounds down to 24, and 51 hours rounds down to 48).

It is not essential that a FREQ_OUT command appear in the control file. If it doesn't, the output frequency defaults to 12 hours, which is also the simulation time step. Nor must the FREQ_OUT command appear first in the control file. It can appear anywhere before the TIME_PHASE command.

It matters, however, whether the FREQ_OUT command occurs before or after the OUT_SPEC commands. As I discuss later, FREQ_OUT commands can also appear in the input files named in the OUT_SPEC commands (the files with extenders .OUT in the above listing of MASTER). The FREQ_OUT command in the control file applies to all output files, while each other FREQ_OUT command applies to only its own output file. If the FREQ_OUT command in the control file occurs before the OUT_SPEC commands, it will be superseded for any individual output file that has its own FREQ_OUT command. Conversely, if it occurs after the OUT_SPEC commands, then it will supersede any output frequencies specified for individual output files.

THE END COMMAND

When LDM encounters the END key word, it terminates. This line can be omitted, for if LDM encounters an end-of-file while reading the control file, it will also terminate. In either case, the message "Execution terminated: NORMAL END" will be displayed on the screen.

THE ORDER OF THE COMMANDS

The order of the commands in the control file should be as shown above. The rules are:

- FREQ_OUT is optional, and if it occurs it may appear anywhere before the TIME_PHASE command. But it may have a different effect depending on whether it occurs before or after an OUT_SPEC command. See the appropriate section above.
- There must be exactly one ATTRITION command. Except for the FREQ_OUT command, if any, the ATTRITION command should appear first.
- There must be at least one SUPPORT command, and it (or they, if there is more than one) should appear next.
- There must be one or more OUT_SPEC commands. They should follow the SUPPORT commands.
- There must be exactly one TIME_PHASE command, and it must appear after all the OUT_SPEC commands. This command starts the simulation process, so any command encountered after this one has no effect on the simulation.
- END must appear last, just after the TIME_PHASE command. Anything that follows it will be ignored.

POSSIBLE ERRORS

If LDM encounters a record with something in columns 1–8 that has not been mentioned in this section, it will write this message to the log file:

KEY WORD IN CONTROL FILE NOT RECOGNIZED

and this message to the screen:

Execution Terminated: KEY WORD UNRECOGNIZABLE

The record containing the offending key word will be echoed to the screen and the log file just before this message appears. To recover from this error, the user need only decide which key word he really intended and spell it correctly in the control file.

It is also possible, though unlikely, that LDM will be unable to read one of the records in the control file. (In the previous error, LDM could read the record but did not recognize the key word.) In this case, LDM will write the following error messages:

Log : READ ERROR #(errno) IN CONTROL FILE

Screen: Execution terminated: READ ERROR

In the message to the log file, (errno) is an error number assigned by the RM/FORTRAN compiler. If this error occurs late enough—e.g., while LDM is reading the END record—the case may have run successfully in spite of the error.

4. THE ATTRITION FILE

The ATTRITION file contains data for the combat support module of LDM. These data govern the FLOT movement, attrition, and resource consumption calculated in each simulated time period. These data are established during calibration, and the user will generally have little reason to change them.

The examples in this section are taken from the ATTRITION file for the test case, which is named ATT.DAT.

TITLES

The ATTRITION file may (but need not) begin with one or more title records. These records can each contain up to 72 characters of text, and they will be echoed to the screen and to all the standard output files. (They will not be echoed to the log file.) LDM will read and echo each record in the ATTRITION file, until it finds a record that begins with a recognizable key word.

KEY WORDS RECOGNIZED IN THE ATTRITION FILE

LDM recognizes the following key words when it encounters them in the ATTRITION file. If the first eight columns of a record contain the first eight characters of any of these key words, LDM will stop echoing records and will consider further records to contain data.

CATEGORIES
POSTURE
MOVEMENT
ATTRITION ... END
CBT_LOSS ... DEP ... END

The key words ATTRITION and CBT_LOSS have subordinate key words associated with them. Once LDM reads either the key word ATTRITION or CBT_LOSS, it will not recognize the other key words until after it has encountered END. It recognizes the key word DEP only after it has read CBT_LOSS but before it encounters the accompanying END.

CATEGORIES

LDM does not simply throw all the forces into a single homogeneous engagement. Rather, each side's engaged force is spread across engagement categories with a variety of ratios of Blue to Red combat worth.[1] There are 16 combat worth ratio categories, and they are defined by the CATEGORIES data. These data are entered in the following five input lines.

```
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
--------------------------------------------------------------------------
CATEGORIES
0.1     0.15    0.2     0.25    0.333   0.5     0.667   1.0     RATLIM
1.5     2.0     3.0     4.0     5.0     6.667   10.0    9999.9  RATLIM
0.1     0.1225  0.1732  0.2236  0.2885  0.4080  0.5775  0.8167  RATAVG
1.2247  1.7321  2.4495  3.4641  4.4721  5.7736  8.1652  10.0    RATAVG
--------------------------------------------------------------------------
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
```

The first line contains the key word CATEGORIES starting in column 1. The next four lines are grouped into two pairs. At eight numbers per line, each pair of lines specifies 16 numbers, one for each combat worth ratio category. The format for a pair of lines is:

First of Two Lines		Second of Two Lines	
Combat Worth Category	Columns	Combat Worth Category	Columns
1	1– 8	9	1– 8
2	9–16	10	9–16
3	17–24	11	17–24
4	25–32	12	25–32
5	33–40	13	33–40
6	41–48	14	41–48
7	49–56	15	49–56
8	57–64	16	57–64

These numbers define ratios of Blue-to-Red combat worth at which various functions (such as attrition functions) will be evaluated. The pair of lines labeled RATLIM (extreme right) define the 16 combat

[1]LDM uses a combat worth function to measure the overall combat power of their forces. Coefficients in this function specify the relative worth of different weapons on the two sides—e.g., a Blue tank vs. a Red tank, or a Blue tank vs. a Blue Armored Personnel Carrier (APC).

worth ratios at which successive engagement categories begin and end. For example, the first engagement category includes forces engaged at a Blue-to-Red ratio of less than 0.1, the second category includes ratios between 0.1 and 0.15, etc. The pair of lines labeled RATAVG specify the combat worth ratios that will be considered typical of each category. For example, LDM will consider all engagements in the first category to have combat worth ratios of exactly 0.1, in the second category ratios of exactly 0.1225, and so on. Attrition functions, FLOT movement functions, etc., will all be evaluated at these points.

POSTURE

The next section of data is headed by the key word POSTURE. The POSTURE data occupy seven lines, as follows:

```
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
------------------------------------------------------------------------
POSTURE
1.0     1.0     0.9998  0.9994  0.998   0.9883  0.934   0.6964  RED ATT
0.2002  0.0327  0.004   0.0004  0.0001  0.      0.      0.
0.      0.      0.0002  0.0006  0.002   0.0112  0.06    0.253   STATIC
0.5516  0.5091  0.3557  0.2174  0.1434  0.0954  0.0563  0.0323
0.      0.      0.      0.      0.      0.0005  0.006   0.0505  BLUE ATT
0.2482  0.4582  0.6403  0.7822  0.8565  0.9046  0.9437  0.9677
------------------------------------------------------------------------
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
```

The Blue and Red forces that engage in combat during any time period are distributed over 16 combat worth ratio categories, as described above. But then, the forces within each category are further distributed among three postures—Red Attack Blue Defend, Static, and Blue Attack Red Defend. The POSTURE data specify the fraction of forces in each combat worth ratio category that adopt each of the three postures. There are three pairs of lines in the posture data, one for each posture. Each pair of lines has 16 coefficients, one for each combat worth ratio category. Their formats are the same as described in CATEGORIES above. The first pair of lines gives the fractions of force in each combat worth ratio category that are in the Red Attack Blue Defend posture. The second and third pairs of lines do the same for the Static and Blue Attack Red Defend postures, respectively.

The reader may find it easier to understand the organization of these data if I rearrange them into tabular form and add the CATEGORIES data:

Category	Range of Combat Worth	Average Combat Worth	Frac. of Engaged Combat Worth in: Red Att Blue Def	Static	Blue Att Red Def
1	0.0 – 0.1	0.1	1.0	0.	0.
2	0.1 – 0.15	0.1225	1.0	0.	0.
3	0.15 – 0.2	0.1732	0.9998	0.0002	0.
4	0.2 – 0.25	0.2236	0.9994	0.0006	0.
5	0.25 – 0.333	0.2885	0.998	0.002	0.
6	0.333 – 0.5	0.4080	0.9883	0.0112	0.0005
7	0.5 – 0.667	0.5775	0.934	0.06	0.006
8	0.667 – 1.0	0.8167	0.6964	0.253	0.0505
9	1.0 – 1.5	1.2247	0.2002	0.5516	0.2482
10	1.5 – 2.0	1.7321	0.0327	0.5091	0.4582
11	2.0 – 3.0	2.4495	0.004	0.3557	0.6403
12	3.0 – 4.0	3.4641	0.0004	0.2174	0.7822
13	4.0 – 5.0	4.4721	0.0001	0.1434	0.8565
14	5.0 – 6.667	5.7736	0.	0.0954	0.9046
15	6.667 – 10.0	8.1652	0.	0.0563	0.9437
16	10.0 – 9999.9	10.0	0.	0.0323	0.9677

From the table, one sees that of all forces engaged at combat worth ratios between 0.0 and 0.1 (category 1), 100 percent will be in a Red Attack Blue Defend posture. Similarly, of forces engaged at ratios between 1.5 and 2.0 (category 10), 3.27 percent will be in a Red Attack Blue Defend posture, 50.91 percent in a Static posture, and 45.82 percent in a Blue Attack Red Defend posture.

MOVEMENT

The ATTRITION file also contains data specifying FLOT movement for each engagement category and each side. Movement data occupy 13 input lines, as follows:

```
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
--------------------------------------------------------------------------
MOVEMENT
-23.3267-23.1157-20.679 -15.5314-9.5658 -4.5328 -2.3897 -1.4697 B/RA
0.       0.      0.      0.      0.      0.      0.      0.
0.       0.      0.      0.      0.      0.      0.      0.       B/ST
0.       0.      0.      0.      0.      0.      0.      0.
-23.3267-23.1157-20.679 -15.5314-9.5658 -4.5328 -2.3897 -1.4697 B/BA
1.4697  2.3897  4.5328  9.5658  15.5314 20.679  23.1157 23.3267
21.3947 21.2012 18.9663 14.245  8.7735  4.1574  2.1918  1.348   R/RA
-1.348  -2.1918 -4.1574 -8.7735 -14.245 -18.9663-21.2012-21.3947
0.       0.      0.      0.      0.      0.      0.      0.       R/ST
0.       0.      0.      0.      0.      0.      0.      0.
0.       0.      0.      0.      0.      0.      0.      0.       R/BA
-1.348  -2.1918 -4.1574 -8.7735 -14.245 -18.9663-21.2012-21.3947
--------------------------------------------------------------------------
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
```

The first line contains the key word MOVEMENT starting in column 1. The next 12 lines consist of six pairs of lines containing sixteen numbers per pair. Their formats are the same as described in CATEGORIES above. These numbers define six functions, with 16 numbers for each function, relating the movement of an average unit of either Blue or Red to the ratio of Blue-to-Red combat worth, for engagements of a given posture. Movement is in kilometers per 12 hours. Positive numbers indicate movement favorable to Blue, negative numbers movement favorable to Red.

The two lines labeled B/RA give the movement of the average Blue unit in engagements in which Red is attacking and Blue defending. The B/ST lines give Blue movement in static engagements. The B/BA lines give Blue movement in engagements in which Blue is attacking and Red defending. The remaining six lines give Red unit movement for engagements with the same three postures.

ATTRITION . . . END

Also in the ATTRITION file are the combat worth coefficients and attrition functions for each weapon. Here is a sample of these data:

```
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
-------------------------------------------------------------------------
ATTRITION
BRIGADE TANK    READY          0.1     0.1
0.31    0.3     0.285   0.275  0.26    0.235   0.2     0.155   B/RA
0.095   0.055   0.03    0.01   0.005   0.      0.      0.
0.0007  0.0007  0.0007  0.0007 0.0007  0.0007  0.0007  0.0007  B/ST
0.0007  0.0007  0.0007  0.0007 0.0007  0.0007  0.0007  0.0007
0.0765  0.075   0.0725  0.0705 0.069   0.066   0.0635  0.061   B/BA
0.058   0.055   0.0525  0.05   0.048   0.046   0.0435  0.042
0.005   0.077   0.117   0.14   0.16    0.195   0.217   0.244   R/RA
0.27    0.292   0.314   0.335  0.35    0.365   0.382   0.395
0.      0.      0.      0.0005 0.0018  0.0048  0.009   0.0145  R/ST
0.0245  0.036   0.046   0.052  0.055   0.057   0.0586  0.059
0.      0.005   0.015   0.025  0.04    0.06    0.085   0.113   R/BA
0.14    0.155   0.165   0.172  0.177   0.18    0.183   0.186
BRIGADE APC     READY          0.059   0.078
  .
  .
  .
END
-------------------------------------------------------------------------
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
```

The first line contains the key word ATTRITION starting in column 1. Following this are 13 lines for each weapon. The first line of each group of 13 (the second line of the example) contains the name of a weapon in columns 1–24. The name consists of three 8-character fields and matches the name of an *activity* defined in the SUPPORT input file (see Sec. 5). Thus, these inputs define an interface between the combat module and the support module of LDM. The weapon name is followed on the same input line by two numbers in columns 33–40 and columns 41–48. The numbers are the combat worth per weapon of this type to Blue (first number) and Red (second number).

Then come six pairs of lines defining attrition of Blue and Red weapons of this type as a function of Blue-to-Red combat worth ratio and engagement posture. Their formats are the same as described in CATEGORIES above. These data define six functions in the following order:

1. Attrition of Blue weapons in Red Attack Blue Defend engagements;
2. Attrition of Blue weapons in Static engagements;
3. Attrition of Blue weapons in Blue Attack Red Defend engagements;
4. Attrition of Red weapons in Red Attack Blue Defend engagements;
5. Attrition of Red weapons in Static engagements; and
6. Attrition of Red weapons in Blue Attack Red Defend engagements.

Each coefficient specifies the fraction of engaged weapons that will be hit during a 12-hour period. This fraction includes all hits, whether catastrophic or minor.

Following these lines will be another 13 records with the same formats for the next weapon, followed by another 13 records for the third weapon, and so on until attrition functions have been provided for every weapon. Following the last record for the last weapon, there should be a record with the key word END starting in column 1.

CBT_LOSS . . . DEP . . . END

Finally, the ATTRITION file contains CBT_LOSS data. CBT_LOSS data describe what happens to resources during engagements. When a tank (the weapon) is hit, for example, CBT_LOSS data specify how many of its crew become casualties, how much ammunition is lost or consumed,[2] and what fractions of the tanks that are hit become catastrophic kills, mobility kills, or firepower kills. A sample is shown here:

```
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
--------------------------------------------------------------------------
CBT_LOSS



  DEP   BRIGADE TANK    K-KILL
        BRIGADE TANK    READY   0.0     0.3     0.0     0.3
  DEP   BRIGADE TANK    F-KILL
        BRIGADE TANK    READY   0.0     0.15925 0.0     0.15925
  DEP   BRIGADE TANK    M-KILL
        BRIGADE TANK    READY   0.0     0.47775 0.0     0.47775
  .
  .
  DEP   BRIGADE OTHR AMMNET LOSS
        BRIGADE TANK    READY   0.0     25.0    0.0     2.8
        BRIGADE HELO    READY   39.4    0.0     7.2     0.0
        BRIGADE ATM     READY   0.0     175.0   0.14    4.6
        BRIGADE INFANTRYREADY   0.0     173.0   0.0     0.0
  .
  .
END
--------------------------------------------------------------------------
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
```

The first line contains the key word CBT_LOSS starting in column 1. The next line starts with the key word DEP in columns 3–5, and in columns 9–32 names an activity defined in the SUPPORT input file (see Sec. 5). The rate at which this activity occurs during the current simulated time period is to be estimated with data from the lines that follow. Thus, the rate of this activity is the *dependent variable* (hence the key word DEP) in an estimating equation. These data therefore define another interface between the combat module and the support module, in addition to the one defined by the ATTRITION . . . END data. The format of a DEP line is:

[2]In the test case, ammunition is measured in hundreds of pounds.

Columns		Content
1– 8		The key word DEP. Note that it starts in column 3.
9–16	-- +	Three 8-character alphanumeric fields
17–24	|	that match the name of an activity defined
25–32	-- +	in the SUPPORT input file.

Following a DEP line is a group of one or more lines with blanks in the first eight columns. These lines provide the data for estimating the dependent variable identified in the preceding DEP line. Columns 9–32 specify a weapon, which must have been previously identified as a weapon in the ATTRITION data described immediately above. Columns 33–64 contain four coefficients in the estimating equation, as follows:

Columns		Content
1– 8		(blank).
9–16	-- +	Three 8-character alphanumeric fields
17–24	|	that match the name of a weapon (see the
25–32	-- +	ATTRITION . . . END section above).
33–40		Amount of Blue resource lost or evacuated per Blue weapon of this type engaged.
41–48		Amount of Blue resource lost or evacuated per Blue weapon of this type hit.
49–56		Amount of Red resource lost or evacuated per Red weapon of this type engaged.
57–64		Amount of Red resource lost or evacuated per Red weapon of this type hit.

An example may make it clear how the CBT_LOSS data are used. The first two records listed above, after the key word CBT_LOSS, declare that the number of tanks catastrophically killed, BRIGADE .. TANK .. K-KILL, is equal to, for Blue, 0.0 times the number of the weapon BRIGADE .. TANK .. READY engaged plus 0.3 times the number of the same weapon hit. The first DEP record identifies BRIGADE .. TANK .. K-KILL as the dependent variable. The record following the DEP record identifies BRIGADE .. TANK .. READY as the weapon that provides the engaged and hit numbers. This record also contains the two coefficients 0.0 and 0.3. The remaining two coefficients define the corresponding CBT_LOSS function for Red.

The end of this section is marked by a line containing the key word END starting in column 1.

POSSIBLE ERRORS

When LDM encounters an input record that it fails to recognize, it writes an error message to the log file and another one to the screen. The following errors can occur while the model is reading the ATTRITION file.

The first error message occurs if LDM has read all the records in the ATTRITION file but expects more data.

Log : UNEXPECTED END-OF-FILE READING ATTRITION DATA

Screen: Execution terminated: EARLY EOF

The next error occurs if the user tries to define more weapons than LDM has allocated space for.

Log : TOO MANY WEAPONS. MAX POSSIBLE IS 20

Screen: Execution terminated: TOO MANY WEAPONS

The next error will occur if LDM is reading CBT_LOSS data and encounters a weapon name that was not defined in the previously read ATTRITION data. This would happen, for example, if the data in the ATTRITION file are out of order and LDM reads CBT_LOSS data before it finds ATTRITION data. Remember, the ATTRITION data must precede the CBT_LOSS data. Or the weapon name might be misspelled in either the ATTRITION data or the CBT_LOSS data.

Log : NO MATCH FOR THE FOLLOWING WEAPON IN **CBT_LOSS** DATA:
(image of record with unmatched weapon)

Screen: Execution terminated: UNMATCHED WEAPON IN **CBT_LOSS** DATA

The next error occurs if the user tries to define more CBT_LOSS functions than LDM has allocated space for. There is one CBT_LOSS function for each record with the key word DEP, and as currently dimensioned, LDM allows only 60 functions.

Log : TOO MANY CBT_LOSS FUNCTIONS. MAX POSSIBLE IS 60

Screen: Execution terminated: TOO MANY CBT_LOSS FUNCTIONS

The next error can occur only if the SUPPORT files (see Sec. 5) are read before the ATTRITION file. Thus, if the user follows the directions given in Sec. 3 for the order of the commands in the control file, LDM should never generate this error while reading the ATTRITION file. (LDM can generate this error, however, while reading the SUPPORT files.)

Log : TOO MANY ACTIVITIES. MAX POSSIBLE IS 500

Screen: Execution terminated: TOO MANY ACTIVITIES

Finally, if a record in the ATTRITION file has the wrong format (e.g., character data in a field where LDM expects numerical data), LDM will generate the following messages:

Log : READ ERROR #(errno) IN **ATTRITION** FILE

Screen: Execution terminated: READ ERROR

In the message to the log file, (errno) is an error number assigned by the RM/FORTRAN compiler. The following situation would generate an error of this type. LDM expects four records after the key word CATEGORIES before it encounters another key word. If there are fewer, it will encounter the next key word when it is expecting numerical data. Since alphanumeric data cannot be read with a strictly numeric format, an error will be generated.

5. THE SUPPORT FILES

The SUPPORT files (there must be one or more) provide data to the *support* module of LDM. They name the *resources* considered in a particular run of LDM and define the *activities* simulated in the support module. These activities are defined by their effects on the resources (e.g., consumption, production, transport, transformation).

The SUPPORT files will be established during the calibration process (see Ref. [1], Secs. 2 and 3), and the user will change them only if he wishes to analyze a change in the theater support structure. Such a change might be as small as a change in repair or transportation times, or it could be as large as adding an entire logistics function such as POL distribution (see Ref. [1], Sec. 6).

The examples in this section are taken from the three SUPPORT files for the test case, BATTLE.DAT, MAINT.DAT, and PER_SUP.DAT.

TITLES AND THE KEY WORD SUPPORT

As in all the input files, the user may (but need not) use the first several records for titles or descriptive information. These records may contain up to 72 characters each. LDM will read each record and echo it to all the standard output files. When it reads a record with the key word SUPPORT in columns 1–8, it will begin interpreting the following records as data. For example, the SUPPORT file named PER_SUP.DAT begins as follows:

```
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
------------------------------------------------------------------------
Activities that make people and supplies available for use
in combat.
SUPPORT
------------------------------------------------------------------------
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
```

KEY WORDS RECOGNIZED IN THE SUPPORT FILES

LDM will recognize the following key words when it encounters them in the first eight columns of a SUPPORT file record.

```
SUPPORT
STOCK    ... RHS
CONSTR   ... RHS
PIPE     ... FROM ... TO ... COEF
END
```

The key word SUPPORT was identified above as the signal that all following records would contain data rather than titles. The primary key words STOCK and CONSTR both have the subordinate key word RHS,[1] while PIPE has subordinate key words FROM, TO, and COEF. A record with a primary key word (e.g., CONSTR), plus all records with subordinate key words (e.g., RHS) that follow immediately, form a single group. But the subordinate key words have to be appropriate to the primary key word, or LDM will report an error. For example, LDM will report an error if a PIPE record is followed by an RHS record, or a STOCK record is followed by a FROM, TO, or COEF record.

RESOURCES: STOCK . . . RHS AND CONSTR . . . RHS

In LDM there are two kinds of resources that constrain the rates of the support activities. "Stock" resources are quantities that can be carried over from one time period to the next. For example, tank ammunition on hand at Corps Support Command (COSCOM) at the end of one time period is still there to start the next. "Constraint" resources are not carried over. For example, if the manhours or truck hours available for transporting ammunition in one period are not fully utilized, the unused portions are lost forever. They cannot be saved for use in the next period.

Groups of records starting with the key words STOCK . . . RHS or CONSTR . . . RHS describe how the amounts of these resources should be calculated. LDM saves these data in a *resource matrix*, whose content and use is fully discussed in Ref. [1], App. B.

[1]For right-hand side.

STOCK and CONSTR Records

For LDM to recognize a resource of either kind, that resource must be given a name. There are several ways to do this, but the most straightforward is to name the resource in a STOCK or CONSTR record—STOCK for a "stock" resource, and CONSTR for a "constraint" resource. Here is an example of each kind of record:

```
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
-----------------------------------------------------------------------
STOCK   COSCOM    TANK AMM  ON HAND
CONSTR  COSCOM    TRCK HRS  AVAIL
-----------------------------------------------------------------------
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
```

The format for these records is:

Columns		Content
1– 8		The key word STOCK or CONSTR.
9–16	--+	Three 8-character alphanumeric
19–26	|	fields constitute the name of a
29–36	--+	resource.

The three data elements in columns 9–36 form the name of a resource. The user is entirely free to invent his own naming convention for resources (and for activities as well). My own convention is the following.

Location: The first field, in columns 9–16, identifies the location or echelon at which this resource is found. Example entries might be BRIGADE, DISCOM, COSCOM, and THEATER. Clearly, any given generic resource (e.g., artillery ammunition) will be found at many locations. In LDM, however, it will be deemed a different resource at each location. One such resource can, of course, be transformed into another by a transportation activity.

Generic Name: The second field, in columns 19–26, contains the generic name of the resource. Examples are TANK, ARTY AMM (artillery ammunition), WPN CREW, and in the example above, TRCK HRS (truck hours).

Status: The third field, in columns 29–36, contains the condition or status of the resource. For example, a weapon crewman at DISCOM might be ON HAND, meaning he is available for duty,

or he might be in WAIT DSP, meaning that he is awaiting disposition.

RHS Records

Any number of RHS records can be associated with each STOCK or CONSTR record. Each RHS record defines a contribution to the amount of the resource named in the associated STOCK or CONSTR record. That contribution, which takes place at the start of each simulated time period, depends on the resource named in the RHS record. Here is an example.

```
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
-------------------------------------------------------------------------
CONSTR  COSCOM    TRCK HRS  AVAIL
  RHS   COSCOM    TRUCK     ON HAND   12.0            12.0
-------------------------------------------------------------------------
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
```

In this example, the resource defined by the CONSTR record is the number of truck hours available to transport things from the COSCOM to other echelons. This capacity is proportional to the number of trucks at COSCOM, and an RHS record is used to specify the constant of proportionality. In the example, the coefficient 12.0 is the proportionality constant, and it may be interpreted as the number of hours per 12-hour period that a truck is available. Thus, if the user wished to increase the transportation capacity at COSCOM by, say, 120 hours per 12-hour simulated time period, he would add ten trucks—that is, ten of the resource COSCOM .. TRUCK .. ON HAND. (Adding the resource is done in the TIME_PHASE file—see Sec. 7.) The format for an RHS record is:

Columns		Content
1– 8		The key word RHS. Note that RHS starts in column 3. This indents these records for better readability.
9–16	--+	
19–26	|	Name of resource.
29–36	--+	

39–46	Coefficient to be used by Blue to multiply the amount of the resource identified in columns 9–36.
55–62	Coefficient for Red.

Only a STOCK resource may be named in an RHS record. CONSTR resources are not allowed. If a CONSTR resource name appears in an RHS record, LDM will generate an error message and terminate execution.

Resource Type Defaults to STOCK

As described below, resources can be named in other kinds of records than STOCK and CONSTR. If the user intends the resource to be a "stock" resource and there is no need for RHS records to describe how the amount of the resource shall be calculated, then the STOCK record is optional for that resource. It may appear, if the user wishes, or it may be omitted.

By contrast, every "constraint" resource requires a CONSTR record, whether or not it needs any accompanying RHS records. And even a "stock" resource needs a STOCK record if it needs accompanying RHS records.

SUPPORT ACTIVITIES: PIPE . . . FROM . . . TO . . . COEF

Also in the file are groups of records that define support activities. Records in these groups start with key words PIPE, FROM, TO, and COEF. LDM saves data on activities in an *activity matrix*, whose content and use are fully discussed in Ref. [1], App. B.

Network Structure of the Support System

The support structure in LDM can be represented, in large part, as a network. Figure 6 shows the part of the network for the test case representing the four activities that affect ammunition. The support structure contains a duplicate of this network for each type of ammunition simulated in the test case, including tank ammunition, artillery ammunition, and other ammunition. The four activities are:

1. RESUPPLY of ammunition to the COSCOM, which transports ammunition from THEATER to COSCOM.
2. RESUPPLY of ammunition to the DISCOM, which transports ammunition from COSCOM to DISCOM.
3. RESUPPLY of ammunition to the BRIGADE, which transports ammunition from DISCOM to BRIGADE.
4. NET LOSS, which takes ammunition from the BRIGADE but does not deliver it to any other node. The ammunition is considered consumed or destroyed, and it is dropped from the simulation.

Each node in the network represents a stock of some resource. Each link represents an activity that transforms the resource represented by the origin node to the resource represented by the destination node. To illustrate, in Fig. 6 consider the RESUPPLY link from the ON HAND node at COSCOM to the ON HAND node at DISCOM. The link represents the transportation activity that moves ammunition from COSCOM to DISCOM. In the test case, this activity is implemented by the following records:

```
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
-----------------------------------------------------------------------
PIPE    DISCOM    TANK AMM  RESUPPLY  12.0    15.3    12.0    15.3
  FROM  COSCOM    TANK AMM  ON HAND   1.0             1.0
  TO    DISCOM    TANK AMM  ON HAND   1.0             1.0
  FROM  DISCOM    TANK AMM  REORDER   1.0             1.0
  COEF  COSCOM    ORD HRS   AVAIL     0.01419         0.01419
  COEF  COSCOM    TRCK HRS  AVAIL     0.12            0.12
  COEF  COSCOM    DRVR HRS  AVAIL     0.24            0.24
-----------------------------------------------------------------------
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
```

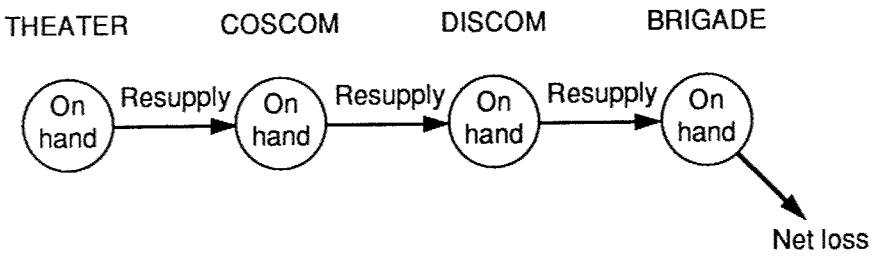

Fig. 6—Activity Network for Ammunition

These records define one activity for Blue and one for Red. The first number in each record (the first two numbers in the PIPE record) defines the Blue activity, while the second number (the last two in the PIPE record) defines the Red activity. The numbers for Blue and Red may differ, but the activity names and the resources they affect (identified in the FROM, TO, and COEF records) must be the same.

The first record in this example starts with the key word PIPE. A PIPE record always marks the beginning of a group of records that defines an activity. Here is its format:

Columns		Content
1– 8		The key word PIPE.
9–16	--+	
19–26	|	Name of activity.
29–36	--+	
39–46		Average time required for Blue to perform this activity.
47–54		Blue priority for this activity.
55–62		Average time required for Red to perform this activity.
63–70		Red priority for this activity.

Several kinds of records may follow a PIPE record. These specify the nodes from which and to which the corresponding link goes and the amounts of the constraint resources that the activity uses. The format of these records is:

Columns		Content
1– 8		Key word FROM, TO, or COEF. The key words all start in column 3. This indents the records for better readability.
9–16	--+	
19–26	|	Name of resource.
29–36	--+	
39–46		Coefficient to be used by Blue to multiply the amount of the resource named in columns 9–36.
55–62		Coefficient for Red.

In this example, the first FROM record identifies the node from which the link starts (COSCOM .. TANK AMM .. ON HAND), and the TO record identifies the node at which it ends (DISCOM .. TANK AMM .. ON HAND). The remaining records correspond to considerations that cannot be shown conveniently in a network. The second FROM record specifies that a unit of ammunition must be subtracted from the REORDER quantity—i.e., the amount of ammunition the DISCOM is allowed to request from higher echelons—whenever a unit of ammunition is shipped to DISCOM. The first COEF record specifies that the shipment of one unit of ammunition from COSCOM requires 0.01419 manhours from an ordnance person (for example, to load the ammunition on a truck). (In the test case, ammunition is expressed in units of 100 pounds.) The second COEF record specifies that 0.12 truck hours are used to transport each unit of ammunition. (Recall from the earlier example of the CONSTR and RHS records how the total available truck hours are calculated.) The third COEF record specifies that 0.24 manhours of a driver's time is spent per unit of ammunition transported. If any of these resources is in short supply, it may limit the amount of ammunition that can be shipped.

The key words FROM and COEF have identical effects, and the user may freely substitute one for the other. An earlier version of LDM treated FROM and COEF records differently. Although I have eliminated the difference in the current version of LDM, I have retained both key words.

The key word TO, however, is treated differently. LDM delivers resources *to* nodes designated in TO records, whereas it takes resources *from* nodes designated in FROM and COEF records. In addition, it delays the delivery of some of the resource to the TO node so that the average time for delivery is the time specified in the PIPE record.

Time Delays

To simulate time delays, LDM treats each link of the support system network as a pipeline, in which stocks of a resource can be held for a time before delivery. LDM accomplishes this by creating a resource for each activity—in effect, a new node in the activity network—with the same name as the activity. (For this reason, one should not give a resource the same name as an activity.) The time required for the resource to traverse the pipeline is represented by only part of the stock exiting the pipeline during the current time period, the rest being withheld for later periods. The fraction withheld is adjusted so that it takes the desired average time for the resource to pass through the pipeline.[2]

Figure 7 shows, for several different average times to perform the activity, the fraction of the resource that will have been delivered as a function of the amount of time that has passed. In calculating these curves, LDM assumes that the resource is taken from the origin node at a uniform rate, in this example over a period of 12 hours (one simulation cycle). As one should expect, for short average delivery times (e.g., the 6-hour curve) most deliveries are made in the first and second 12-hour cycles. But for longer average delivery times, the deliveries are spread over many cycles. For example, when the average delivery time is 108 hours (the bottom curve), about one-third of the deliveries are made within the first 48 hours. In compensation, some deliveries are delayed well beyond the average time.

Blue and Red Priorities

Two data elements in each PIPE record have yet to be discussed: the Blue priority (columns 47–54) and the Red priority (columns 63–70). Activities may have negative, zero, or positive priorities. LDM calculates the rates of activities with zero priorities in the combat module, using the CBT_LOSS data from the ATTRITION files. Rates of activities with nonzero priorities (negative or positive) are calculated in the support module. Activity rates are calculated in priority order. Thus, the simulation cycle first invokes the support module to compute rates for activities with negative priorities, then invokes the combat module to compute rates for activities with zero priorities, and finally invokes the support module a second time to compute rates for the activities with positive priorities.

[2]In mathematical terms, I have assumed that the time to traverse the pipeline has an exponential distribution. See Ref. [1], App. B for a detailed discussion.

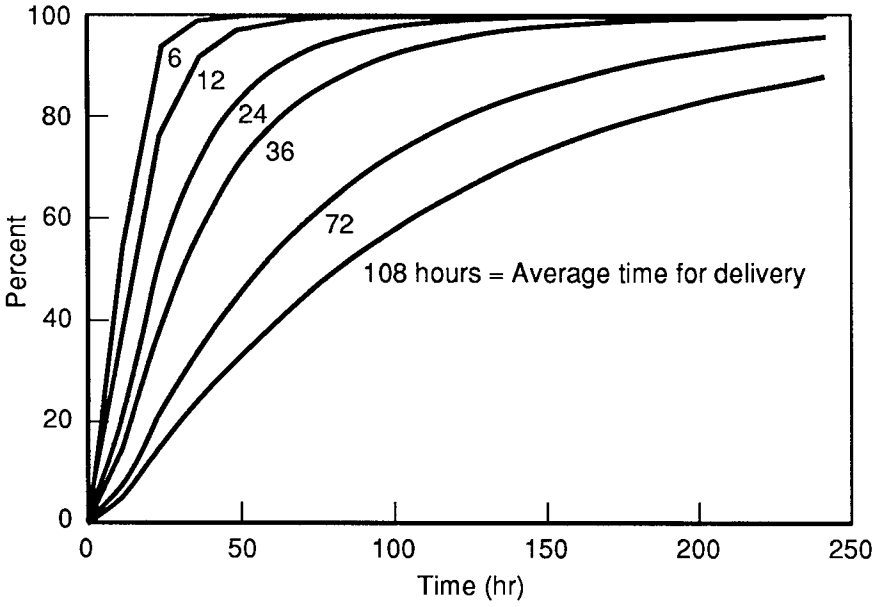

Fig. 7—Percentage of Deliveries by Time

In the test case (see Sec. 8), the activities with negative priorities determine how many weapons of each type can be made available for combat from the resources on hand at the BRIGADE echelon. Activities with zero priorities determine the amount of each type of ammunition consumed in combat, the numbers of casualties suffered, and the equipment lost or damaged. Activities with positive priorities evacuate wounded people and damaged equipment from the BRIGADE echelon, hospitalize the wounded people and repair the damaged equipment, and resupply the BRIGADE with the resources it has lost and evacuated.

An activity may have a unique nonzero priority, or it may share its priority with one or more other activities. When an activity has its own unique priority, LDM will make its rate as large as possible, given the available quantities of the resources it consumes. If the example activity above were the only one with a priority of 15.3 (in fact, it is not), then LDM would make its rate during a simulated time period just large enough that one of the resources it consumes would be depleted. That is, *either*:

- Tank ammunition on hand at COSCOM (COSCOM .. TANK AMM .. ON HAND) is exhausted; *or*
- DISCOM has requisitioned as much as authorized (DISCOM .. TANK AMM .. REORDER); *or*
- The ordnance personnel at COSCOM are handling as much ammunition as they can (COSCOM .. ORD HRS .. AVAIL); *or*
- Trucks at COSCOM are transporting all they can (COSCOM .. TRCK HRS .. AVAIL); *or*
- Truck drivers at COSCOM are driving as much as they can (COSCOM .. DRVR HRS .. AVAIL).

This activity increases the amount of ammunition on hand at DISCOM. Because the resource DISCOM .. TANK AMM .. ON HAND is not depleted by this activity, it cannot be the factor that limits the activity rate.

When two or more activities have the same priority, they share whatever resources they consume in common, rather than one of the activities being able to claim as much of the resource as it can use. The rates of all the activities in the group are increased together, according to a scheme described in Ref. [1], App. B, until each activity in the group is prevented by the depletion of at least one resource from increasing further. In the test case, the DISCOM .. TANK AMM .. RESUPPLY activity shares the priority 15.3 with the corresponding activities for the other ammunition types, ARTY AMM and OTHR AMM. The three activities also share the available ordnance personnel capacity, truck capacity, and driver capacity. Because they share the priority, when one of these capacities is the factor that limits these activity rates, some of each kind of ammunition will be delivered.[3] If each activity had its own unique priority, the one with the earliest priority might consume all of the limiting capacity, leaving none for resupplying the other two ammunition types.

The priority order of the activities can be important. For example, consider two of the resupply activities in Fig. 6, say resupply from THEATER to COSCOM and resupply from COSCOM to DISCOM. The first of these activities delivers ammunition to the COSCOM, while the second issues it from the COSCOM. If the first activity has the earlier priority, then more ammunition will be available for issue from COSCOM, and (depending on the available capacities and allowed reorder quantity) more ammunition may actually be issued from COSCOM.

[3]See Ref. [1], App. B, for a detailed explanation of how the limiting resource is apportioned among activities with shared priorities.

THE END RECORD

If LDM encounters a record with the key word END starting in column 1, it will stop reading from the current SUPPORT file and return to read the next record in the control file (Sec. 3). Any records following END will be ignored.

It is not necessary to include an END record in a SUPPORT file. If there is no END record, LDM will continue reading from the file until it encounters an end-of-file. Then it will behave precisely as if it had encountered an END record at that point.

MULTIPLE SUPPORT FILES

If the user wishes, all the resources and activities may be defined in a single SUPPORT file. Generally, however, the resulting file would be inconveniently large, so LDM allows the user to split the data among any number of SUPPORT files. In the test problem distributed with LDM, the support data are split among BATTLE.DAT, MAINT.DAT, and PER_SUP.DAT.

- BATTLE.DAT contains activities that make weapon systems available for combat and that assess battle damage to equipment and consumption of resources during combat.
- MAINT.DAT contains activities that repair and transport equipment resources so they will be available for use in combat.
- PER_SUP.DAT contains activities that make people and supplies available for use in combat.

Each SUPPORT file may (but need not) begin with titles, but the start of data in that file *must* be signaled by a SUPPORT record. Each SUPPORT file may (but need not) end with an END record.

Some care must be exercised in choosing where to split the support data. A group of associated records must remain together and not be split between SUPPORT files. In particular, RHS records that follow a STOCK or CONSTR record must be in the same file with the STOCK or CONSTR record. Similarly, if a PIPE record is followed by a collection of TO, FROM, and COEF records, those records must remain in the same file as the PIPE record.

POSSIBLE ERRORS

Beware of duplicate names. Make sure that no two activities have the same name. Make sure that no two resources have the same name and that no activity has the same name as any resource. LDM does not generate an error message in such cases.

LDM can generate the following errors while reading the SUPPORT files. First, in the SUPPORT file an RHS record must be in a CONSTR or STOCK group.

Log : THE SUPPORT RECORD SHOWN BELOW MUST FOLLOW A 'STOCK' OR 'CONSTR' RECORD:
(image of record in question)

Screen: Execution terminated: SUPPORT RECORD OUT OF ORDER

Similarly, TO, FROM, and COEF records must be in a PIPE group.

Log : THE SUPPORT RECORD SHOWN BELOW MUST FOLLOW A 'PIPE' RECORD:
(image of record in question)

Screen: Execution terminated: SUPPORT RECORD OUT OF ORDER

If the key word in a SUPPORT record is not in the list at the start of this section, LDM will report that the record cannot be recognized. Typically, this is caused by a misspelled key word, which is easily fixed.

Log : CANNOT RECOGNIZE SUPPORT RECORD SHOWN BELOW:
(image of record in question)

Screen: Execution terminated: BAD SUPPORT RECORD

The user can present a problem to LDM that is too large in one of several possible ways. The only way to fix these problems is to reformulate the problem so that it is smaller in the dimension that has been exceeded. For example, it can have too many resources.

Log : TOO MANY CONSTRAINTS. MAX POSSIBLE IS 1200

Screen: Execution terminated: TOO MANY CONSTRAINTS

Or a problem can have too many activities.

Log : TOO MANY ACTIVITIES. MAX POSSIBLE IS 500

Screen: Execution terminated: TOO MANY ACTIVITIES

It can have too many nonzero coefficients in the resource matrix (see Sec. 9).

Log : TOO MANY RESOURCE MATRIX ENTRIES. MAX POSSIBLE IS 3000

Screen: Execution terminated: TOO MANY RESOURCE MATRIX ENTRIES

Finally, it can have too many nonzero coefficients in the activity matrix (see Ref. [1], App. B).

Log : TOO MANY ACTIVITY MATRIX ENTRIES. MAX POSSIBLE IS 3000

Screen: Execution terminated: TOO MANY ACTIVITY MATRIX ENTRIES

Any support activity with no positive coefficients in the activity matrix consumes no resources. (If it has any strictly negative coefficients, it will even produce resources.) LDM issues the following error message for such activities.

Log : ACTIVITY ** (name of activity) ** HAS NO POSITIVE MATRIX ENTRIES
(message repeated up to 11 times, for different activities)

(number) ACTIVITIES FOUND WITH NO POSITIVE MATRIX ENTRIES

Screen: Execution terminated: BAD ACTIVITIES OR RHS RECORDS

For technical reasons, no RHS record is permitted to name a CONSTR resource. If the user inadvertently does so, LDM will issue the following error message:

```
Log   : FOUND **RHS** RECORDS THAT NAME **CONSTR** RESOURCES, AS FOLLOWS:
                CONSTR    (name of resource)            -+
                  RHS     (name of offending resource)   |
                                  .                      |
                                  .                      | Up to 10
                                                         | CONSTR
                                 OR                      | plus
                                                         | STOCK
                STOCK     (name of resource)             | blocks
                  RHS     (name of offending resource)   |
                                  .                      |
                                  .                     -+
```

Finally, if a record in a SUPPORT file has the wrong format (e.g., character data in a field where LDM expects numerical data), LDM will generate the following messages:

Log : READ ERROR #(errno) IN **SUPPORT** FILE

Screen: Execution terminated: READ ERROR

In the message to the log file, (errno) is an error number assigned by the RM/FORTRAN compiler. The user will be able to determine which SUPPORT file contains the error, because the above message will be immediately preceded in the log file by an image of the corresponding SUPPORT record in the control file.

6. THE OUT_SPEC FILES

The OUT_SPEC files (mnemonic for "output specification") specify which quantities to collect for output. There can be up to ten OUT_SPEC files, each specifying what quantities will appear in a different output file.[1] For each output file, the user specifies whether to collect the specified quantities from every simulated time period, or from every two, or every three, etc., periods. In addition to the quantities specified, LDM will write to each output file the simulated time at which the quantities are output, and the FLOT movement since the previous time the quantities were output.

TITLES AND THE KEY WORD OUT_SPEC

The first several records in an OUT_SPEC file may be used for titles or descriptive information, at the user's discretion. LDM reads 72 characters from each title record and echoes them to the corresponding output file. LDM will write these title lines only to the output file that corresponds to this OUT_SPEC file—the output file named on the same control file command as the OUT_SPEC file (see Sec. 3). (The titles in other kinds of input files are written to all output files.) When LDM reads a record with the key word OUT_SPEC starting in column 1, it will interpret all remaining records as data. In the data for the test case, for example, the OUT_SPEC file named AMMO.OUT begins thus:

```
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
------------------------------------------------------------------------
Output All Flows and Stocks of Ammunition for both sides
OUT_SPEC
------------------------------------------------------------------------
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
```

[1] In the test case distributed with LDM, the OUT_SPEC files are those with the extenders .OUT, while the output files are the files with extenders .PRN. See the description of the OUT_SPEC command in Sec. 3.

KEY WORDS RECOGNIZED IN OUT_SPEC FILES

LDM will recognize the following key words when it encounters them in the first eight columns of an OUT_SPEC record:

OUT_SPEC
FREQ_OUT
COMBAT
BLUE
RED
END

The key word OUT_SPEC was identified above as the signal that all the remaining records would contain data rather than titles. The functions of the other key words are described below.

THE KEY WORD FREQ_OUT

The FREQ_OUT record specifies the output frequency for the corresponding output file only. (Other output files may have their own output frequency specifications.) The record contains the key word FREQ_OUT in columns 1–8 and the number of hours between output reports in columns 13–18. The user must either include the decimal point or right-justify the number. LDM will calculate the number of simulated 12-hour time periods per output cycle from the hours between output reports as follows: If the number of hours between reports is less than 24, LDM will produce output every period. If it is greater than or equal to 24 hours, LDM will round the number down to the next lower multiple of 12 hours. Thus an entry of 28 hours will be rounded down to 24 hours and will result in an output cycle consisting of two periods. Similarly, 51 hours becomes 48 hours and yields an output cycle of four periods. LDM echoes the FREQ_OUT record to the output file that corresponds to this OUT_SPEC file but not to any other output files.

THE KEY WORD COMBAT

As mentioned earlier, LDM consists of a combat module and a support module. Most of the available output options cause various quantities calculated in the support module to be saved for later output. The COMBAT option causes the following three quantities generated by the combat module to be output for each weapon system on each side:

$AWPN_{ni}$ = number of weapons of type n *available* to side i (i = 1 for Blue, i = 2 for Red).

$EWPN_{ni}$ = number of weapons of type n *engaged* on side i.

$HITS_{ni}$ = number of weapons of type n *suffering combat damage* on side i.

If a FREQ_OUT record has specified that the output cycle should consist of more than one simulated time period (that the COMBAT output should occur only every second, third, or nth simulated time period), LDM will accumulate these three quantities over the n periods and will output their sum.

As stated earlier, FLOT movement is automatically written to every output file, including the one collecting outputs for the COMBAT option. LDM will accumulate FLOT movement over all the periods in an output cycle and will output the sum.

The user may elect not to use the COMBAT option; but if he does, he must devote one OUT_SPEC file to this option alone. LDM will ignore any other output quantities specified there. In the test case, the file CBTLIM.OUT is devoted to this purpose, and a listing of this file appears here:

```
Output Combat Limitations for both sides
OUT_SPEC
COMBAT
END
```

THE KEY WORDS BLUE AND RED

Output Options Available

The key words BLUE and RED require additional information to be interpreted fully. Here are some sample BLUE records.

```
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
--------------------------------------------------------------------------
BLUE      INITIAL   BRIGADE   TANK      AUTH
BLUE      INITIAL   BRIGADE   TANK      ON HAND
BLUE      INITIAL   BRIGADE   TANK      RETURNS
BLUE      ACTIVITY  BRIGADE   TANK      NET LOSS
BLUE      ACTIVITY  BRIGADE   TANK      RETURNS
--------------------------------------------------------------------------
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
```

Each record contains five data elements identifying one desired output quantity.

Columns		Content
1–8		BLUE or RED, the side to which this quantity applies.
11–18		Key word identifying kind of quantity to output.
21–28	- +	
31–38	|	Name of resource or activity.
41–48	- +	

The key words that LDM recognizes in columns 11–18 of these records are in the following list:

Option (Key Word)	Use with	Option (Key Word)	Use with
INHERIT	RESOURCE	REMAIN	RESOURCE
EXOG	RESOURCE	UNUSED	RESOURCE
INITIAL	RESOURCE	MINIMUM	RESOURCE
R_INCR	RESOURCE	ACTIVITY	ACTIVITY
AVAIL	RESOURCE	A_INCR	RESOURCE

If the key word ACTIVITY appears in columns 11–18, then columns 21–48 *must* contain the name of an activity, not a resource. The other key words may be used with either a resource or an activity.[2]

The LDM Calculation Cycle

To fully appreciate the quantities that can be output with these options, the reader must be familiar with the LDM calculation cycle. It is described in full in Ref. [1], Apps. A and B, but a brief description appears here. With each calculation cycle, LDM simulates one 12-hour time period. The cycle consists of the following steps:

STEP 1: CALCULATE INHERITED RESOURCES. LDM first calculates the amount of each resource inherited from the previous period. For "constraint" resources, this quantity is always zero, since unused "constraint" resources are not carried forward from one period to the next (recall the distinction made in Sec. 5 between "stock" and "constraint" resources). For "stock" resources, these quantities are taken to be the amount remaining at the end of the previous simulated time period for every period but the first. In the first period the inherited quantities are taken to be zero.

[2]Strictly speaking, these other key words may be used only with a resource. However, as described in Sec. 5, when LDM establishes an activity, it creates an accompanying resource. For example, if the activity involves repairing tanks, LDM will create a resource to keep track of how many tanks are in repair at any time.

$$\text{INHERIT}_i(P) = \begin{cases} \text{REMAIN}_i(P-1) & \text{if i is a "stock" resource} \\ 0 & \text{if i is a "constraint" resource} \end{cases} \quad (1)$$

where:

$\text{INHERIT}_i(P)$ = amount of resource i inherited for period P.

$\text{REMAIN}_i(P-1)$ = amount of resource i remaining from period P – 1.

STEP 2: ADD RESOURCES FROM OUTSIDE THE SYSTEM. Next LDM adds quantities of resources from outside the simulation.

$$\text{INITIAL}_i(P) = \text{INHERIT}_i(P) + \text{EXOG}_i(P) \quad (2)$$

where:

$\text{INITIAL}_i(P)$ = amount of resource i on hand at the start of period P.

$\text{EXOG}_i(P)$ = amount of resource i brought in from outside the system for period P (from the TIME_PHASE file; see Sec. 7).

STEP 3: APPLY THE RESOURCE MATRIX. The next step is to apply the *resource matrix* to the initial resource quantities. (The resource matrix was first mentioned in Sec. 5 and is fully described in Ref. [1], App. B.)

$$\text{AVAIL}_i(P) = \sum_k (\text{INITIAL}_k(P) \times r_{ki}) \quad (3)$$

where:

$\text{AVAIL}_i(P)$ = amount of resource i available for use by the activities during the current period.

r_{ki} = element in row k, column i of the resource matrix.

STEP 4: COMPUTE ACTIVITY RATES. Finally, LDM computes activity rates, and adjusts the resource quantities using the *activity*

matrix. (The activity matrix was first mentioned in Sec. 5 and is fully described in Ref. [1], App.B.)

$$\text{REMAIN}_i\,(P) = \text{AVAIL}_i\,(P) - \Sigma\,(a_{ij} \times \text{ACT}_j\,(P)) \qquad (4)$$

where:

$\text{ACT}_j\,(P)$ = rate of activity j during period P.

a_{aj} = element in row i, column j of the activity matrix.

This brings the calculations full circle and leaves LDM ready to begin the calculation cycle for the next simulated time period.

Descriptions of Output Options

INHERIT: This key word causes LDM to output the amount of the resource named in columns 21–48 that was inherited from the previous output cycle. If the output cycle consists of more than one period,[3] the value from the first period is output and the values from succeeding periods are ignored. That is, in terms of the quantities defined in Eqs. (1)–(4):

$$\text{OUTPUT} = \text{INHERIT}_i\,(P1)$$

where P1 denotes the first period in the output cycle.

This output is not useful for a constraint resource, because it is always zero. For a stock resource, the quantity output in response to this option is the same as the quantity of a stock resource remaining at the end of the previous period, which can be obtained with the REMAIN option.

EXOG: This key word causes LDM to output the amount of a resource added from outside the system during all the periods in the output cycle. That is:

$$\text{OUTPUT} = \Sigma\,\text{EXOG}_i\,(P)$$

where the sum is taken over all periods in the output cycle.

These quantities are inputs to the simulation from the TIME_PHASE file (see Sec. 7), so specifying them as outputs tells the user nothing

[3]Here, and throughout the remainder of this section, "period" means "simulated 12-hour time period." Each period, then, corresponds to one calculation cycle.

he does not already know. However, it does place these inputs alongside other output quantities, so the user can easily relate them.

INITIAL: In response to this key word, LDM outputs the amount of the named resource available at the start of the current output cycle. This is the sum of the inherited amount plus the amount added from outside the simulation during the *first* period of an output cycle only. That is:

$$\text{OUTPUT} = \text{INITIAL}_i\ (\text{P1})$$

R_INCR: In response to this key word, LDM outputs the increment to a resource quantity generated by calculations with the resource matrix (see Eq. (3)), as follows:

$$\text{OUTPUT} = \Sigma\ [\text{AVAIL}_i\ (\text{P}) - \text{INITIAL}_i\ (\text{P})]$$

The sum is taken over all periods in an output cycle.

AVAIL: This output option produces the total amount of a resource available to be consumed by activities during the output cycle.

$$\text{OUTPUT} = \Sigma\ \text{AVAIL}_i\ (\text{P})$$

The sum is taken over all periods in the output cycle.

REMAIN: LDM responds to this key word by outputting the quantity of the named resource that remains at the end of the output cycle. The quantity output is the amount of resource remaining after the last period in the output cycle. That is:

$$\text{OUTPUT} = \text{REMAIN}_i\ (\text{Pn})$$

where Pn denotes the last period in the output cycle.

This option is designed for use with a stock resource. It represents the amount of the resource not used during the current output cycle that will be carried over for use in the next cycle. Since constraint resources are not carried over from one cycle to the next, this option is not particularly relevant for them.

UNUSED: This key word causes LDM to output the sum of the quantities of the named resource remaining after every period in the output cycle. That is:

$$\text{OUTPUT} = \Sigma\ \text{REMAIN}_i\ (\text{P})$$

where the sum is taken over all periods in the output cycle.

This option is designed for use with constraint resources, for which it is a reasonable measure of excess capacity in the system. If some of a constraint resource (e.g., maintenance manhours) is left over at the end of a period, it is lost, but a stock resource is carried over to the next period if it is not used during the current one. That is, the remaining stock is not unused; its use is deferred.

MINIMUM: This key word also causes LDM to output the remaining quantity of the named resource, but the value output is the minimum value encountered during any period in the output cycle. For example, if the output cycle consists of three periods and the maintenance manhours remaining at the DISCOM at the end of the three simulation cycles are 1000, 0, and 2500, respectively, then this key word will cause LDM to output the value 0, the smallest of the three. The formula is:

$$\text{OUTPUT} = \text{Min}\{\ \text{REMAIN}_i\ (P)\ \mid\ \text{all periods P in output cycle}\}$$

This output is designed to indicate whether a particular resource has limited the level of any activity during any period in the output cycle. If the minimum remaining quantity of a resource is zero or negative, that resource limits the levels of all activities that consume that resource. Adding some of that resource to the simulation will allow more of those activities to occur. On the other hand, if the minimum value is positive, it has not limited any activity levels during this output cycle, so adding more of that resource should not alter the simulation results.

ACTIVITY: In response to this key word, LDM outputs the activity level for the named activity, summed over all simulation cycles in the output cycle. That is:

$$\text{OUTPUT} = \Sigma\ \text{ACT}_j\ (P)$$

A_INCR: This key word causes LDM to output the incremental resource produced or consumed during calculations with the activity matrix (see Eq. (4)). The formula is:

$$\text{OUTPUT} = \Sigma\ [\text{AVAIL}_i\ (P) - \text{REMAIN}_i\ (P)]$$

where the sum is taken over all periods in an output cycle. One can interpret this as the net amount of the resource consumed by all the support activities. A positive value, therefore, indicates that more was consumed than produced (e.g., a stockpile has been drawn down), and a negative value that more was produced than consumed (the stockpile has been replenished).

THE KEY WORD END

If LDM encounters a record with the key word END starting in column 1, it will stop reading from the current SUPPORT file and return to read the next record in the control file (Sec. 3). Any records following END will be ignored.

It is not necessary to include an END record in an OUT_SPEC file. If there is no END record, LDM will continue reading from the file until it encounters an end-of-file. Then it will behave precisely as if it had encountered an END record at that point.

POSSIBLE ERRORS

LDM can generate the following error messages while it attempts to read OUT_SPEC data. First, LDM can accommodate only ten output files. If you attempt to define more, LDM will generate the following error.

Log : (number) OUTPUT FILES ARE ALREADY DEFINED.
CANNOT ACCOMMODATE
(image of OUT_SPEC command)

Screen: Execution terminated: TOO MANY OUTPUT FILES REQUESTED

Once all the titles plus an OUT_SPEC record have been read, if any remaining record does not contain FREQ_OUT, BLUE, RED, COMBAT, or END in columns 1–8, LDM will write the next error message. If this happens, check columns 1–8 of these records for spelling and for extraneous characters.

Log : KEY UNRECOGNIZABLE
THE FOLLOWING OUT_SPEC RECORD NOT RECOGNIZED
(image of record in question)

Screen: KEY UNRECOGNIZABLE
Execution terminated: BAD OUT_SPEC RECORD

If the first field contains either BLUE or RED, then an error in columns 11–18 can also provoke an error message. Columns 11–18 must contain INHERIT, EXOG, INITIAL, R_INCR, AVAIL, REMAIN, UNUSED, MINIMUM, ACTIVITY, or A_INCR. Anything else will result in the following message. Again, if you encounter this message, check for a spelling error.

Log : OUTPUT TYPE UNRECOGNIZABLE
THE FOLLOWING OUT_SPEC RECORD NOT RECOGNIZED
(image of record in question)

Screen: OUTPUT TYPE UNRECOGNIZABLE
Execution terminated: BAD OUT_SPEC RECORD

If the first field is BLUE or RED, and the second is ACTIVITY, then columns 21–48 must contain a valid activity name. LDM will search the list of activity names from the SUPPORT and ATTRITION files, and if it canot match one with the name in columns 21–48, it generates the following message. This usually results from an error in spelling the activity name.

Log : CANNOT FIND ACTIVITY NAME
THE FOLLOWING OUT_SPEC RECORD NOT RECOGNIZED
(image of record in question)

Screen: CANNOT FIND ACTIVITY NAME
Execution terminated: BAD OUT_SPEC RECORD

If the first field is BLUE or RED, and the second is anything except ACTIVITY (but is recognized), then columns 21–48 must contain a valid constraint name. LDM will try to match the name in columns 21–48 with a name from the list of constraint names in the SUPPORT files. If it fails, it generates the following message. This usually results from an error in spelling the constraint name.

Log : CANNOT FIND CONSTRAINT NAME
THE FOLLOWING OUT_SPEC RECORD NOT RECOGNIZED
(image of record in question)

Screen: CANNOT FIND CONSTRAINT NAME
Execution terminated: BAD OUT_SPEC RECORD

There is a limit on the number of output quantities that can be specified for any output file. If the limit is exceeded, you must delete some of the records in that OUT_SPEC file, or move them to a different OUT_SPEC file. The error message is:

Log : MORE THAN 220 OUTPUT QUANTITIES
REQUESTED FOR OUTPUT FILE (name of file)

Screen: Execution terminated: EXCESSIVE OUTPUT REQUEST

Finally, if a record in an OUT_SPEC file has the wrong format (e.g., character data in a field where LDM expects numerical data), LDM will generate the following messages:

Log : READ ERROR #(errno) IN **OUT_SPEC** FILE

Screen: Execution terminated: READ ERROR

In the message to the log file, (errno) is an error number assigned by the RM/FORTRAN compiler. The user will be able to determine which OUT_SPEC file contains the error, because the above message will be immediately preceded in the log file by an image of the corresponding OUT_SPEC record in the control file.

7. THE TIME_PHASE FILE

The TIME_PHASE file contains time-specific data. A small amount of the time-specific data consists of time-dependent calibration parameters that will generally be unchanged from case to case, but most of it specifies the amounts of resources to enter the simulation at various times. (The model deals with resources initially present by causing them to enter the simulation at the first simulated time.) Any resource may be added that has been defined previously in a SUPPORT file.

It is these data that the user will generally change from one LDM case to another. For example, suppose the user wishes to investigate the effects of increasing maintenance capacity at DISCOM. He would first run a case with the nominal TIME_PHASE file and then a second case in which an increment of the appropriate resource had been added to the TIME_PHASE file. (In Sec. 5, I pointed out that the appropriate resource for maintenance capacity at the DISCOM was DISCOM .. MNT PSNL .. ON HAND.) Comparing the outcomes of the two cases would provide several measures of the effect of the incremental capacity.

TITLES AND THE KEY WORD TIME_PHASE

Like the other input files, this one can begin with an arbitrary number of title lines, with up to 72 characters each, that will be echoed to all the standard output files. When LDM encounters a line with the key word TIME_PHASE starting in column 1, it interprets all remaining lines as data. To illustrate this, here are the first few lines of the TIME_PHASE file for the test case (called NOMTIME.DAT):

```
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
------------------------------------------------------------------------
Time Specific Data Unmodified from Test Case
TIME_PHASE
------------------------------------------------------------------------
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
```

KEY WORDS RECOGNIZED IN THE TIME_PHASE FILE

LDM recognizes only three key words in the TIME_PHASE file. They are:

TIME_PHASE
TIME
END

The key word TIME_PHASE (only the first eight characters matter) was identified above as the signal that all remaining records in the file would contain data rather than titles.

TIME RECORDS

The record following the key word TIME_PHASE should begin with the key word TIME in column 1. Here is an example:

```
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
-------------------------------------------------------------------------
TIME                          0.      0.5721  1.0     1.0
-------------------------------------------------------------------------
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
```

Columns	Content
1– 8	Key word TIME.
25–32	Time in hours (either right-justify or include a decimal point).
33–40	The "standard deviation," a calibration parameter describing how widely the engaged combat worth ratio varies from one part of the FLOT to another (see Ref. [1], App. A).
41–48	Fraction of Blue weapons available that are actually engaged.
49–56	Fraction of Red weapons available that are actually engaged.

The data in the TIME records are established during calibration, which is described in [1]. You should not change these data without an understanding of the material in Ref. [1], App. A.

RESOURCE INCREMENT RECORDS

Any number of resource increment records can follow a TIME record. (Zero records are allowed, as well as one, two, or any other positive number.) These records specify the amounts of resources to add to the system at the time specified by the most recently encountered TIME record. One may add any resource that has been defined in any of the SUPPORT files (see Sec. 5). Here are some example resource increment records:

```
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
-----------------------------------------------------------------------
BRIGADE TANK     AUTH          100        0
DISCOM  MNT PSNLONHAND          10        0
-----------------------------------------------------------------------
123456789_123456789_123456789_123456789_123456789_123456789_123456789_
```

The format for a resource increment record is:

Columns		Content
1– 8	--+	
9–16	|	Resource name.
17–24	--+	
25–32		Amount of resource to add to Blue. Either right-justify the number or include a decimal point.
33–40		Amount of resource to add to Red.

The first example record adds 100 tanks to Blue's authorization for tanks at the BRIGADE echelon. The second example adds ten Blue maintenance personnel at DISCOM. Neither example adds any resources to Red.

THE END RECORD

If LDM encounters a record with the key word END starting in column 1, it will stop reading from the TIME_PHASE file and return to read the next record in the control file (Sec. 3). Any records following END will be ignored.

POSSIBLE ERRORS

Except for the titles, every record in the TIME_PHASE file must have either a recognized key word in columns 1–8 or a recognized resource name in columns 1–24. Recognized resource names are names of resources defined in the SUPPORT files. If LDM encounters a record that does not meet these criteria, it generates the following message. Usually, the problem will be a misspelled key word or resource name.

Log : COULD NOT PROCESS THE FOLLOWING
TIME_PHASE RECORD:
(image of record in question)

Screen: Execution terminated: BAD TIME_PHASE RECORD

Finally, if a record in the TIME_PHASE file has the wrong format (e.g., character data in a field where LDM expects numerical data), LDM will generate the following messages:

Log : READ ERROR #(errno) IN **TIME_PHASE** FILE

Screen: Execution terminated: READ ERROR

In the message to the log file, (errno) is an error number assigned by the RM/FORTRAN compiler.

8. THE TEST CASE

The test case distributed with LDM illustrates how the LDM inputs are used to define the problem to be simulated. The LDM user should be able to make modest modifications to the test case files to generate problems of his own design. If the user wishes to make more radical modifications to the test case, or to design his own problems from scratch, he should refer to Ref. [1]. It will help to have listings of the test case input files at hand. These are the ATTRITION file (ATT.DAT), the SUPPORT files (BATTLE.DAT, MAINT.DAT, and PER_SUP.DAT), the TIME_PHASE file (NOMTIME.DAT), and the OUT_SPEC files (CBTLIM.OUT, LOSSES.OUT, BDERES.OUT, and AMMO.OUT).

THE LDM SIMULATION CYCLE

Figure 8 shows the steps in the LDM simulation cycle, and identifies which input files provide data for each step. Starting at the top, the model first sets the time to signal the start of a new time period (its basic cycle is 12 hours) and reads the amounts of the various resources entering the theater during that period. Table 1 shows the resources considered in the test case. These resources may appear at any of four echelons or locations: BRIGADE, DISCOM, COSCOM, and THEATER (echelons above Corps). Also, a resource may have more than one possible status (e.g., a TANK can be operational or awaiting repair). When a resource enters the theater, it is necessary to specify at what echelon and in what status it enters.

Next, LDM calculates stock and constraint resources at each echelon and in each status available for use by the support activities during the current time period. (Stock and constraint resources are defined in Sec. 5). The available quantity of a stock resource includes the amount of the resource on hand at the end of the previous period, plus the amount added to the theater during the current period. It also includes the amount due to arrive from other echelons or statuses during the current period. For example, some of the ammunition in transit at the end of the previous period will arrive at its destination during the current period, and this must be added to the available ammunition.

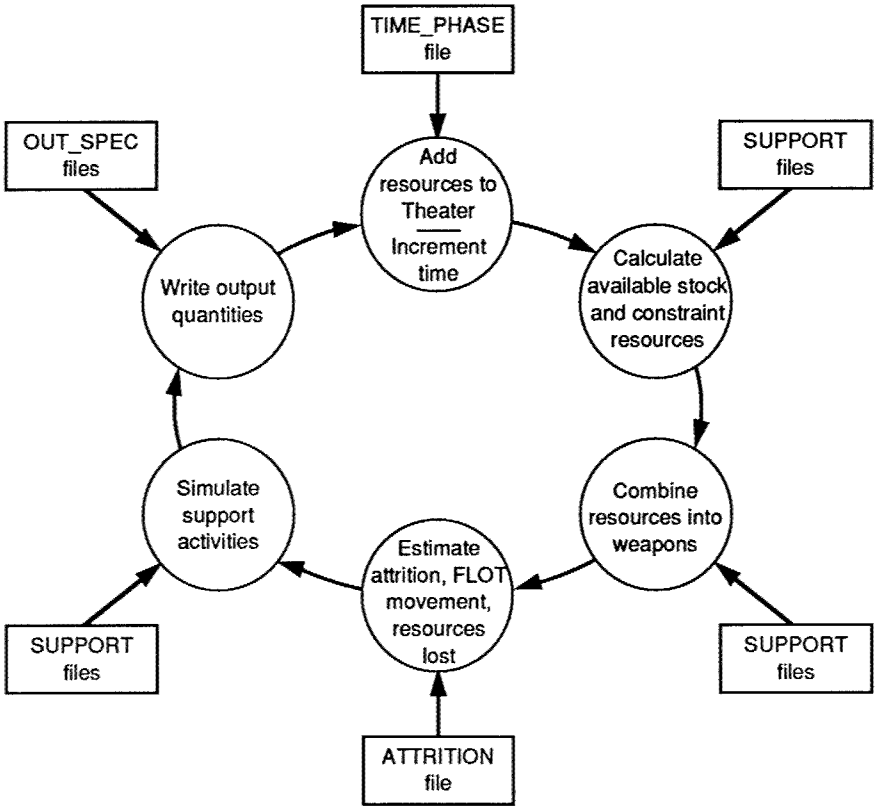

Fig. 8—The LDM Simulation Cycle

A constraint resource typically represents a capacity to perform a support function or a limit on the quantity of a stock resource that one echelon may requisition from others. The available amount of a constraint resource is calculated from the available quantities of the appropriate stock resources. For example, the maintenance capacity at an echelon is calculated from the number of maintenance personnel on hand at that echelon.

In the third step, the available resources are combined into weapon systems available for combat. For example, a tank cannot enter combat unless there is a crew available, plus specified amounts of various types of ammunition. Shortages of resources thus reduce the number of weapon systems available for combat. Available weapon systems are calculated for both Blue and Red. Only the resources in the

Table 1

Resources Considered in the Test Case

Supplies	Personnel	Equipment
	Combat Resources	
Tank ammo	Weapon crew	Tank
Artillery ammo	Combat personnel	APC
Other ammo	Helo crew	Helo
	Artillery personnel	ATM
	Driver	Artillery
		Recovery
		HET
	Support Resources	
	Driver	Truck
	Maint. personnel	
	Medic	
	Supply personnel	
	Ordnance personnel	

COMBAT RESOURCES part of Table 1 are used in weapon systems, and only resources at the BRIGADE are available to be incorporated into weapon systems.

In step four, LDM estimates the outcomes of one 12-hour period of combat between the available Blue and Red weapon systems. The outcomes estimated are:

- The number of each weapon type engaged;
- The number of hits on each weapon type;
- The average distance the FLOT moves; and
- The amounts of the various resources that are lost, consumed, or damaged in various ways.

In step five, LDM simulates the support activities. Activities occur at every echelon and include the recovery, evacuation, and repair of damaged equipment and the resupply of equipment, ammunition, and personnel. The resources in the SUPPORT RESOURCES part of the table are necessary to perform many of the support activities, although they are not generally consumed by those activities. Instead, as described above, LDM computes capacities to perform support activities from the quantities of these resources.

Finally, output quantities are written to the various output files. These quantities are collected throughout the simulation cycle.

STEP 1: ADD RESOURCES TO THE THEATER

At the start of the simulation cycle, resources are added to the theater. Table 1 shows the resources considered in the test case. Each resource may appear at four echelons (BRIGADE, DISCOM, COSCOM, and THEATER) and in any of several statuses. For example, equipment at the DISCOM can be in serviceable status or awaiting repair. The TIME_PHASE file (see Sec. 7) specifies the amount of each resource entering the theater at each echelon in each status during each time period. Both Blue and Red entering resources are specified.

Here is an example of how the TIME_PHASE file specifies these quantities.

```
TIME                          0.     0.5721  1.0      1.0
BRIGADE TANK     ON HAND       3256     4841
BRIGADE APC      ON HAND       5896     5793
```

The first line (with the key word TIME) specifies that the resources named on subsequent lines enter the simulation at time 0. (See Sec. 7 for a discussion of the other three inputs on that line.) The next two lines specify the quantities of TANKs and APCs that enter the simulation at the BRIGADE in ON HAND (operational) status. On each line, the first number specifies the quantity entering on the Blue side, while the second number specifies the Red quantity.

Resources on hand at the start of the simulation are modeled by having them enter the simulation at time zero. Table 2 shows the Blue initial resource quantities for the test case, and Table 3 shows the Red initial quantities. Values in these tables were taken from the file NOMTIME.DAT. At all echelons except THEATER, the TIME_PHASE file specifies both an authorized quantity and a quantity on hand. The authorized quantity limits the amount that LDM will move forward to an echelon in order to replace losses. No authorized quantities are needed at the THEATER, since there is no echelon further back to replace THEATER losses. The quantity on hand is the amount of the resource physically present at the start of the simulation.

Some of the resource quantities in these tables are unreasonably large. For example, Blue is specified to have 99,999,999 units of tank ammunition on hand at the THEATER. (Each unit of ammunition is

Table 2

Blue Resources Initially in Theater

	THEATER	COSCOM		DISCOM		BRIGADE	
	ON HAND	AUTH	ON HAND	AUTH	ON HAND	AUTH	ON HAND
TANK	3378	120	3	94	2	3256	3256
APC	1385	928	0	898	0	5896	5896
HELO	0	8	0	10	0	384	384
ATM	4348	139	35	105	29	2690	2690
ARTY	637	75	9	28	9	721	721
HET	0	99400	99400	406	406	406	406
RECOVERY	0	0	0	0	0	407	407
TRUCK	43614	5926	5926	1328	1328	0	0
WPN CREW	99999	468	0	342	0	56608	56608
CBT PSNL	99999	53	0	35	0	30810	30810
HEL CREW	98988	4	0	10	0	768	768
ART PSNL	99970	404	31	87	82	2812	2812
DRIVER	407910	98350	98350	3768	3768	46045	45259
MNT PSNL	10010	14765	14765	5261	5261	0	0
MEDIC	9885	13387	13387	2124	2124	0	0
SUP PSNL	9910426	9915555	9915555	9993248	9993248	0	0
ORD PSNL	100000	100000	100000	1080	1080	0	0
TANK AMM	99999999	434399	434399	141262	141262	211500	211500
ARTY AMM	99999999	678822	678822	331278	331278	803000	803000
OTHR AMM	99999999	2281531	2281531	771707	771707	99178880	99178880

Table 3

Red Resources Initially in Theater

	THEATER	COSCOM		DISCOM		BRIGADE	
	ON HAND	AUTH	ON HAND	AUTH	ON HAND	AUTH	ON HAND
TANK	558	1043	476	211	96	4841	4841
APC	76	987	493	205	83	5793	5793
HELO	0	44	0	24	0	144	144
ATM	13029	4249	15252	695	3394	9201	9167
ARTY	750	597	379	80	73	2268	2208
HET	0	80	80	45	45	45	45
RECOVERY	0	0	0	0	0	46	46
TRUCK	8326	8880	8880	7497	7497	0	0
WPN CREW	99999	4462	3796	426	0	44020	44020
CBT PSNL	100041	1364	638	483	23	48500	48500
HEL CREW	9963	17	0	17	0	816	816
ART PSNL	100531	572	494	135	110	15644	15644
DRIVER	8404	9025	9025	7633	7633	1000	1000
MNT PSNL	741	25920	25920	5151	5151	0	0
MEDIC	8013	8110	8110	6069	6069	0	0
SUP PSNL	9924000	9998000	9998000	9922593	9922593	0	0
ORD PSNL	9924000	9998000	9998000	9922593	9922593	0	0
TANK AMM	12556380	2000000	2000000	1683254	1683254	408000	408000
ARTY AMM	3000000	3994528	3994528	3363062	3363062	612000	612000
OTHR AMM	8183620	1000000	1000000	842667	842667	99170000	99170000

100 pounds, so this amounts to 5 million tons.) The amounts of these resources were set large so that they would not constrain combat performance. Indeed, one useful analysis technique is to simulate a base case, identify a resource that is in short supply (and hence constrains combat performance), and then create a second case in which a huge initial quantity of that resource is added to the base case.[1] A comparison of results from the second case with results from the base case will indicate both the maximum amount of the resource it would be useful to add and the maximum difference adding the resource could make.

Later additions of resources represent the arrival of new units or resupplies of materiel and personnel replacements in the theater. In the test case, new combat units are added at the BRIGADE. A support "tail" accompanies them and is added at the DISCOM. To represent the arrival of a unit, the TIME_PHASE file contains increments of both resources on hand and resource authorizations. If the authorized resources were not increased, LDM would refuse to replace losses by the new units.

Figure 9 shows the cumulative additions of the TANK resource in the test case for both Blue and Red. There are two curves for each side, one for the authorized TANKs at the BRIGADE and the other for total TANKs on hand delivered to all four echelons. The TANKs at time zero are distributed as shown in Tables 2 and 3. Later TANKs are added mostly at the BRIGADE, with a few added at the DISCOM. Similar curves can be generated for all the other resources from the data in the file NOMTIME.DAT.

Figure 9 shows two differences between Blue and Red. First, Red adds units to his combat force rapidly, more than doubling the number of TANKs authorized at BRIGADE over the ten days of the simulation. By contrast, Blue adds only a few units, increasing his TANK authorizations by perhaps 20 percent. Second, consider the difference between the two curves for each side (the difference between total tanks on hand in the theater and tanks authorized at the BRIGADE). One can interpret this difference as the inventory of war reserve tanks available to replace tank losses. Red provides many fewer war reserve tanks than Blue.

In the test case, no resources enter at the COSCOM or THEATER after time zero. However, a user could add resources at these

[1]To determine whether a resource is limiting, use the MINIMUM output option described in Sec. 6. The REMAIN option (for a stock resource) and UNUSED option (for a constraint resource) can also be used.

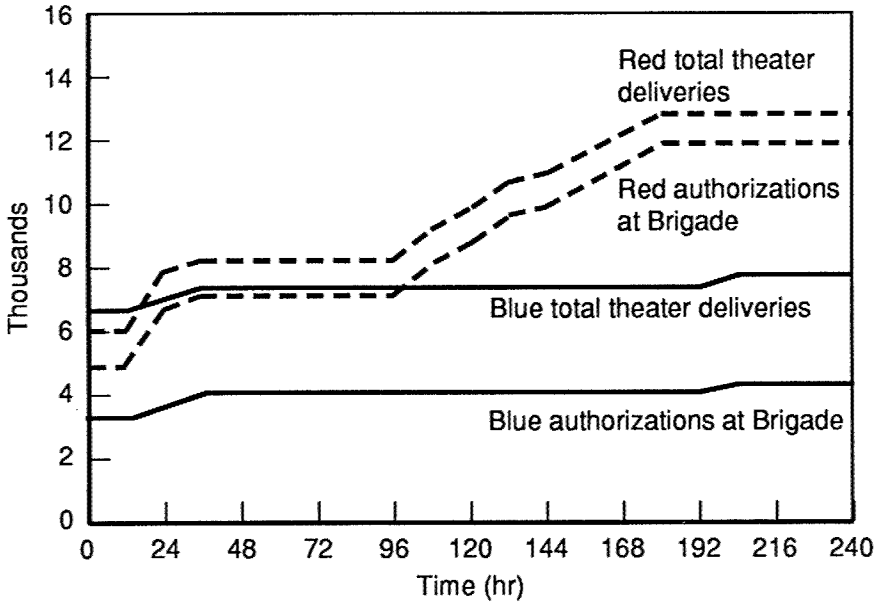

Fig. 9—Authorized and Delivered TANKs

echelons to represent the addition of new CSS units, such as maintenance or transportation units. Resources entering at the THEATER could also include replacement personnel and equipment and ammunition to be used to replace losses and consumption.

STEP 2: CALCULATE AVAILABLE STOCK AND CONSTRAINT RESOURCES

In this step, LDM calculates the stock and constraint resources at each echelon and in each status available for use by the support activities during the current time period. Appendix B of Ref. [1] describes these calculations in detail, and understanding the test case does not require further discussion of how the available stock resources are calculated. However, it is necessary to understand the calculation of constraint resources.

I will defer mention of most of the constraint resources until I discuss the support activities that require them, but as an example I present here the calculation of maintenance capacity at the DISCOM. The relevant SUPPORT records (found in the file MAINT.DAT) are:

```
CONSTR  DISCOM    MMH        REPAIR
  RHS   DISCOM    MNT PSNL   ON HAND   2.85            2.85
```

The CONSTR record defines a constraint resource called DISCOM .. MMH .. REPAIR, which should be interpreted as the repair capacity at DISCOM, measured in maintenance manhours. The RHS record specifies that the available maintenance manhours will be the quantity of MNT PSNL on hand at the DISCOM multiplied by 2.85. That is, I assume that each maintenance person can provide 2.85 productive hours of maintenance in each 12-hour time period.

STEP 3: COMBINE RESOURCES INTO WEAPONS

In the test case, a weapon is defined as a combination of resources. A tank (the weapon system), for example, consists of a tank (the equipment resource) plus a crew and supplies of tank ammunition and other ammunition. Certain activities defined in the SUPPORT files represent the combination of resources into weapons, and the rates of these activities are the numbers of weapons that can be made available for combat. In the test case, these activities are in the file BATTLE.DAT. Table 4 shows the combinations of resources that make up the various Blue weapons. A similar table could be constructed for Red weapons. Quantities of ammunition in the table are expressed in units of hundreds of pounds.

To represent the formation of these weapons in LDM, I define an activity for each column of Table 4. In LDM, activities correspond to columns in a matrix (the *activity* matrix), and resources to rows in the same matrix. Each activity will have a coefficient in a row of the activity matrix for each nonzero entry in the table. In an LDM activity, however, an entry in a row indicates that the activity consumes (if the entry is positive) or produces (if the entry is negative) the corresponding resource. But combining resources into weapon systems does not actually consume those resources. So I define some

Table 4

Resource Combinations That Form Blue Weapons

Resource	Tank	APC	Helo	ATM	Inf	Arty	Recov	Het
TANK	1.0							
APC		1.0						
HELO			1.0					
ATM				1.0				
ARTY						1.0		
RECOVERY							1.0	
HET								1.0
WPN CREW	4.0	4.0						
HEL CREW			2.0					
CBT PSNL				4.0	1.0			
ART PSNL						4.0		
DRIVER							2.0	2.0
TANK AMM	52.6							
OTHR AMM	12.0	16.8	41.0	18.63	1.0			
ARTY AMM						265.0		

constraint resources to be consumed by the formation of weapon systems, while the real resources are left alone. These constraint resources are defined by the following records:

```
CONSTR   BRIGADE    TANK       WEAPON
  RHS    BRIGADE    TANK       ON HAND    1.0              1.0
CONSTR   BRIGADE    TANK AMM   WEAPON
  RHS    BRIGADE    TANK AMM   ON HAND    1.0              1.0
CONSTR   BRIGADE    OTHR AMM   WEAPON
  RHS    BRIGADE    OTHR AMM   ON HAND    1.0              1.0
CONSTR   BRIGADE    WPN CREW   WEAPON
  RHS    BRIGADE    WPN CREW   ON HAND    1.0              1.0
```

Their names are BRIGADE .. TANK .. WEAPON, BRIGADE .. TANK AMM .. WEAPON, etc. The records starting with the key word RHS govern the way LDM calculates the quantities of these resources available at the start of a time period. For example, LDM will set the starting amount of the resource BRIGADE .. TANK .. WEAPON equal to 1.0 multiplied by the amount of the resource BRIGADE .. TANK .. ON HAND.

The activity that calculates the number of tanks available for combat is called BRIGADE .. TANK .. READY. The SUPPORT records that define this activity are:

```
PIPE    BRIGADE   TANK       READY     0.      -10.     0.      -10.
  FROM  BRIGADE   TANK       WEAPON    1.               1.
  COEF  BRIGADE   TANK AMM   WEAPON    52.6             29.2
  COEF  BRIGADE   OTHR AMM   WEAPON    12.0             5.45
  COEF  BRIGADE   WPN CREW   WEAPON    4.0              4.0
```

The first column of numbers in the FROM and COEF records agrees with numbers for the "tank" column in Table 4.[2] The second column of numbers are the corresponding amounts of resources for Red. This activity (and others like it for the other weapons) appears in the file BATTLE.DAT.

The four numbers in the PIPE record are the average times required to execute the activity and the priority of the activity, first for Blue and then for Red. The average times of these activities are zero, meaning that no resources are sequestered in pipelines. All the weapons that can be formed from the resources are immediately available to engage in combat.

The priority of an activity determines when in the simulation cycle LDM calculates its rate and in what order relative to the other activities. For each weapon system, the file BATTLE.DAT contains a block of records like the one above, all at the BRIGADE and all with a READY status. The PIPE records for these blocks are:

```
PIPE    BRIGADE   TANK       READY     0.      -10.     0.      -10.
PIPE    BRIGADE   APC        READY     0.      -10.     0.      -10.
PIPE    BRIGADE   HELO       READY     0.      -9.      0.      -9.
PIPE    BRIGADE   ATM        READY     0.      -8.      0.      -8.
PIPE    BRIGADE   INFANTRY   READY     0.      -7.      0.      -7.
PIPE    BRIGADE   ARTY       READY     0.      -6.      0.      -6.
PIPE    BRIGADE   RECOVERY   READY     0.      -5.      0.      -5.
PIPE    BRIGADE   HET        READY     0.      -5.      0.      -5.
```

These activities all have negative priorities, and this signals LDM to calculate the rates of these activities *before* it invokes the combat

[2]FROM and COEF records have the same effects and can therefore be used interchangeably. This was not true in earlier versions of LDM.

module to estimate attrition, FLOT movement, and resources lost or consumed.[3] Within these activities, those with the earliest (most negative) priorities will be calculated first, and those with later (less negative) priorities will be calculated later. If two or more activities share the same priority, they will be calculated simultaneously.

The order of calculation governs how the activities share resources that more than one activity consumes. For example, it happens (see BATTLE.DAT) that both TANK and APC weapon systems use the WPN CREW resource. Since the activities forming these weapon systems have the same priority (-10), they share the available WPN CREW between them.[4] If one of the weapon systems (e.g., TANK) had an earlier priority, then TANKs would have first call on the available WPN CREW, and only after all possible TANKs were made available for combat would any WPN CREW be allocated to APCs. Similarly, RECOVERY and Heavy Equipment Transporter (HET) share a priority, and also share the resource DRIVER. Thus, even if DRIVER is in short supply, some of both "weapons" will be available for combat.[5]

STEP 4: ESTIMATE ATTRITION, FLOT MOVEMENT, AND RESOURCES LOST

Once all the activities with negative priorities have been calculated, LDM invokes the combat module to estimate attrition, FLOT movement, and resources lost or consumed. The data for these calculations are found mostly in the ATTRITION file (see Sec. 4). (The ATTRITION file for the test case is called ATT.DAT.) The only data found elsewhere are:

---

[3]Activities with priorities of zero correspond to the loss and consumption of resources, calculated at step 4 of the simulation cycle (see Fig. 8). Activities with positive priorities are calculated at step 5.

[4]See Ref. [1], App. B for a detailed discussion of how the shared resource is apportioned among the activities with equal priorities.

[5]The reader may wonder why RECOVERY and HET are considered weapons. After all, they don't shoot, and according to the ATTRITION tables in the ATTRITION file, they have no combat worth and do not suffer combat damage (although one could change this by modifying the ATTRITION file). I reason that recovery vehicles and HETs that are used for battlefield recovery and evacuation are associated with the engaged brigades. That is, if a brigade is not engaged during a time period, its recovery vehicles and HETs will not perform these functions during that period. For the user to calculate the engaged recovery vehicles and HETs (as contrasted with those merely available), they must be defined as weapons.

- The numbers of the Blue and Red weapon systems available for combat. These were calculated in step 3 of the simulation. LDM knows which activities from the SUPPORT file correspond to weapon systems because these activities are named in the ATTRITION data in file ATT.DAT (see Sec. 4);
- The fractions of Blue and Red available weapon systems that actually engage (found in the TIME record of the TIME_PHASE file, Sec. 7); and
- The "standard deviation" of the combat worth ratio distribution (also in the TIME record of the TIME_PHASE file). This parameter determines how widely the ratio of Blue to Red engaged combat worth varies from one part of the FLOT to another.

The calculations performed by LDM's combat module are fully described in Ref. [1], App. A, and one can gain a sense of those calculations from Sec. 4. There will be no more discussion of them here. The quantities calculated are:

- Number of each weapon system engaged;
- Number of each weapon system hit in combat;
- Average movement of the FLOT; and
- Resources lost, consumed, or damaged.

The user calculates the resources lost, consumed, or damaged using the CBT_LOSS data from the ATTRITION file. As described in Sec. 4, the CBT_LOSS data identify certain quantities as *dependent variables*. The name of each dependent variable is the name of one of the activities in the SUPPORT files, whose rate is calculated in LDM's combat module from the numbers of the various weapon systems engaged or hit. These activities can be identified in the SUPPORT files by the fact that they have priorities of zero. For the test case, activities involving loss or damage to equipment can be found in the file BATTLE.DAT. Activities that consume ammunition or that kill or wound personnel can be found in the file PER_SUP.DAT.

STEP 5: SIMULATE SUPPORT ACTIVITIES

Once the rates have been calculated for activities with priority zero, LDM returns to the support module to calculate rates for the remaining activities. These all have positive priorities.

The support activities in the test case implement a rather simple support structure. This structure is best depicted by a network in which

the links correspond to activities and the nodes correspond to stock resources. A link drawn from node 'A' to node 'B' represents an activity that consumes resource 'A' and produces resource 'B'. For example, if resource 'A' is THEATER .. TANK AMM .. ON HAND and resource 'B' is COSCOM .. TANK AMM .. ON HAND, then the link from 'A' to 'B' represents an activity that transports tank ammunition from the THEATER to the COSCOM. Or, if resource 'A' is DISCOM .. TANK .. DS QUEUE and resource 'B' is DISCOM .. TANK .. ON HAND, then the link from 'A' to 'B' represents a tank repair activity at the DISCOM.

The network does not depict all aspects of the support structure. Capacities and other constraint resources do not appear, and a few activities and stock resources (e.g., authorized quantities) fit poorly into the network construct.

The network that depicts the entire support structure for the test case is too large to show as a single figure, so I break it into several figures and discuss each fragment in a separate section below. The fragments are the support structure for ammunition, personnel, equipment recovery and evacuation, and equipment repair and resupply.

The Support Structure for Ammunition

I first discuss the consumption and resupply of ammunition. Figure 10 shows the relevant part of the support structure. The four nodes correspond to stocks of ammunition at the THEATER, COSCOM, DISCOM, and BRIGADE. Three of the links correspond to activities that transport ammunition between adjacent echelons; the fourth

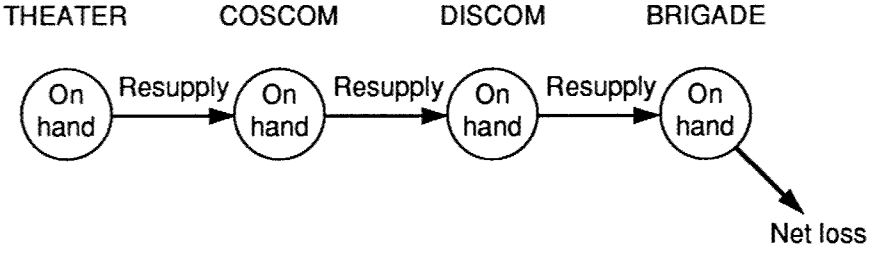

Fig. 10—The Support Structure for Ammunition

link, which has no destination node, represents the loss and consumption of ammunition. For tank ammunition, the four activities are:

```
BRIGADE    TANK AMM  NET LOSS
COSCOM     TANK AMM  RESUPPLY
DISCOM     TANK AMM  RESUPPLY
BRIGADE    TANK AMM  RESUPPLY
```

The resources and activities represented by the nodes and links of this network are replicated for each of the three ammunition types in the test case (see Table 1).

The NET LOSS Activity. The NET LOSS activity represents the amount of that resource lost or consumed in combat. This activity has priority zero, so its rate is calculated in step 4 and uses CBT_LOSS data from the ATTRITION file. For tank ammunition, this activity is implemented by the following records in the PER_SUP.DAT file. There are similar records for the other ammunition types.

```
PIPE     BRIGADE   TANK AMM  NET LOSS  0.0     0.0     0.0     0.0
  FROM   BRIGADE   TANK AMM  ON HAND   1.0             1.0
  FROM   DISCOM    TANK AMM  REORDER   -1.0            -1.0
  FROM   COSCOM    TANK AMM  REORDER   -1.0            -1.0
  FROM   BRIGADE   TANK AMM  REORDER   -1.0            -1.0
```

The first FROM record in this group causes the amount of tank ammunition on hand at the BRIGADE to be reduced by one unit for each unit of ammunition lost or consumed. The other three FROM records cause *increases* (note the coefficients of -1.0) in three constraint resources, identified as REORDER quantities.

The REORDER Constraints. The REORDER constraints govern how much ammunition an echelon is permitted to requisition. LDM

initializes the allowable requisitions for TANK AMM by the BRIGADE with the following records:

```
CONSTR  BRIGADE   TANK AMM  REORDER
  RHS   BRIGADE   TANK AMM  AUTH       1.0              1.0
  RHS   BRIGADE   TANK AMM  ON HAND    -1.0             -1.0
  RHS   BRIGADE   TANK AMM  RESUPPLY   -1.0             -1.0
```

These define the reorder quantity to equal the quantity of tank ammunition authorized by the BRIGADE, less the quantity on hand and the quantity already in the resupply pipeline at the start of the current time period. This initialization takes place in step 2 of the simulation cycle, when the available stock and constraint resources are calculated. I have already mentioned that the calculation of the NET LOSS activity, which occurs in step 4, subsequently increases the allowable requisitions by the amount of ammunition lost or consumed.

There are also REORDER constraints at DISCOM and COSCOM. The following records implement these REORDER constraints. Similar records exist for the other ammunition types.

```
CONSTR  DISCOM    TANK AMM  REORDER
  RHS   BRIGADE   TANK AMM  AUTH       1.0              1.0
  RHS   BRIGADE   TANK AMM  ON HAND    -1.0             -1.0
  RHS   BRIGADE   TANK AMM  RESUPPLY   -1.0             -1.0
  RHS   DISCOM    TANK AMM  AUTH       1.0              1.0

  RHS   DISCOM    TANK AMM  ON HAND    -1.0             -1.0
  RHS   DISCOM    TANK AMM  RESUPPLY   -1.0             -1.0
CONSTR  COSCOM    TANK AMM  REORDER
  RHS   BRIGADE   TANK AMM  AUTH       1.0              1.0
  RHS   BRIGADE   TANK AMM  ON HAND    -1.0             -1.0
  RHS   BRIGADE   TANK AMM  RESUPPLY   -1.0             -1.0
  RHS   DISCOM    TANK AMM  AUTH       1.0              1.0
  RHS   DISCOM    TANK AMM  ON HAND    -1.0             -1.0
  RHS   DISCOM    TANK AMM  RESUPPLY   -1.0             -1.0
  RHS   COSCOM    TANK AMM  AUTH       1.0              1.0
  RHS   COSCOM    TANK AMM  ON HAND    -1.0             -1.0
  RHS   COSCOM    TANK AMM  RESUPPLY   -1.0             -1.0
```

Allowable requisitions at each echelon include the allowable requisitions at all echelons further forward. When I first formulated a

support structure for LDM, I allowed each echelon to requisition just enough to fill its ammunition stocks to its authorized quantity and did not permit the echelon to requisition ammunition on behalf of echelons further forward. In that first formulation, the authorized quantity at the DISCOM was small compared with the average demand per time period at the BRIGADE. But the DISCOM could never requisition more per time period than its authorization and hence could never supply more than a fraction of the BRIGADE demands. To prevent an echelon from becoming a bottleneck in this fashion, I reformulated the REORDER constraints in the manner shown above.

The RESUPPLY Activities. The RESUPPLY activities for tank ammunition are implemented by blocks of records like the following:

```
PIPE    COSCOM    TANK AMM  RESUPPLY  24.0     15.1    24.0     15.1
  FROM  THEATER   TANK AMM  ON HAND   1.0              1.0
  TO    COSCOM    TANK AMM  ON HAND   1.0              1.0
  FROM  COSCOM    TANK AMM  REORDER   1.0              1.0
  COEF  THEATER   ORD HRS   AVAIL     0.01419          0.01419
  COEF  THEATER   TRCK HRS  AVAIL     0.24             0.24
  COEF  THEATER   DRVR HRS  AVAIL     0.48             0.48
```

This activity transports tank ammunition from the THEATER to the COSCOM (see the first FROM record and the TO record). Similar activities, not shown here, move it from COSCOM to DISCOM, and from DISCOM to BRIGADE. Because the activity adds tank ammunition to the stocks at the COSCOM, it must also reduce the allowable requisitions there. This is accomplished by the second FROM record.

The activity also consumes the constraint resources that represent ordnance personnel capacity, truck capacity, and driver capacity. Ordnance personnel are intended to represent the people in DS and GS ordnance companies who "handle" ammunition—i.e., load ammunition on trucks, maintain stockpiles of ammunition, etc. All three

capacities are initialized in step 2 of the simulation cycle. At the THEATER, for example:

```
CONSTR  THEATER   ORD HRS   AVAIL
  RHS   THEATER   ORD PSNL  ON HAND   2.85          2.85
CONSTR  THEATER   TRCK HRS  AVAIL
  RHS   THEATER   TRCK HRS  ON HAND   12.           12.
CONSTR  THEATER   DRVR HRS  AVAIL
  RHS   THEATER   DRVR HRS  ON HAND   2.85          2.85
```

In step 5, when LDM calculates the rate of the above ammunition resupply activity, the three COEF records cause the available amounts of ordnance personnel capacity, truck capacity, and driver capacity to be diminished by the specified amounts.

The rate of the above RESUPPLY activity (the amount of ammunition to be resupplied) is limited by the availability of the resources it consumes. It cannot, for example, consume more ammunition at the THEATER than the THEATER has on hand. Nor can more than the REORDER quantity of ammunition be resupplied. Finally, this activity cannot consume more ordnance personnel capacity, truck capacity, or driver capacity than are available.

Once LDM has calculated the amount of ammunition to be resupplied, it will adjust the quantities of resources consumed or produced accordingly. Thus, ammunition will be taken from the THEATER. Some will be added directly to the ammunition stocks at the COSCOM, while the remainder will be sequestered in the transportation pipeline and will emerge in later time periods. (LDM automatically determines the amounts of ammunition treated in each of these ways, based on the duration of the activity.) The REORDER quantity will be reduced, as will the ordnance personnel capacity, truck capacity, and driver capacity.

Other activities will also consume ordnance personnel capacity and/or truck capacity or driver capacity. Activities whose priorities dictate that they be calculated before this ammunition resupply activity will have consumed some of these capacities already. If by chance they had consumed all the available capacity of one of these kinds, then no ammunition resupply could have taken place. Similarly, activities whose priorities dictate that they be calculated after the ammunition resupply activity can use only whatever amounts of these capacities are left at that point.

The Support Structure for Personnel

Figure 11 shows the network of activities for the personnel resources. It consists of evacuation, medical treatment, and resupply activities at each echelon. For the weapon crew personnel type, the activities are:

```
BRIGADE    WPN CREW  NET LOSS
DISCOM     WPN CREW  MEDEVAC
DISCOM     WPN CREW  IN HOSP
COSCOM     WPN CREW  MEDEVAC
COSCOM     WPN CREW  IN HOSP
THEATER    WPN CREW  MEDEVAC
THEATER    WPN CREW  IN HOSP
COSCOM     WPN CREW  RESUPPLY
DISCOM     WPN CREW  RESUPPLY
BRIGADE    WPN CREW  RESUPPLY
```

The complete set of activities is replicated for the personnel types that are incorporated into weapons in the test case (see the COMBAT RESOURCES portion of Table 1). The personnel types that are not incorporated in any weapon system (the SUPPORT RESOURCE portion of Table 1) are never found at the BRIGADE, so only the

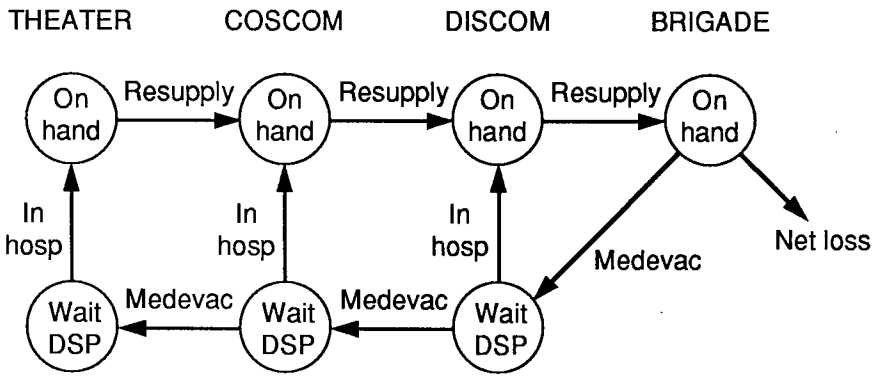

Fig. 11—The Activity Network for Personnel

following truncated set of activities is needed (shown for maintenance personnel only).

```
COSCOM     MNT PSNL  MEDEVAC
COSCOM     MNT PSNL  IN HOSP
THEATER    MNT PSNL  MEDEVAC
THEATER    MNT PSNL  IN HOSP
COSCOM     MNT PSNL  RESUPPLY
DISCOM     MNT PSNL  RESUPPLY
```

The REORDER Constraints. Like ammunition, each combat personnel type has an authorization and a REORDER quantity at each echelon except the THEATER. (Each support personnel type has an authorization and a REORDER quantity at each echelon except the THEATER and BRIGADE.) I do not show the records that define and initialize the personnel REORDER quantities, because they are essentially the same as the ammunition REORDER quantities discussed earlier. (These records can be found in the file PER_SUP.DAT.)

The NET LOSS and MEDEVAC Activities Involving the BRIGADE Echelon. Each combat personnel type has a NET LOSS activity that represents people killed or immediately evacuated from the theater being simulated. It also has a MEDEVAC activity that represents evacuating wounded personnel that will spend some time in hospital in the theater. Both of these activities have priority zero, so their rates are calculated in step 4 of the simulation cycle, using CBT_LOSS data from the ATTRITION file. The NET LOSS activity reduces the number of personnel ON HAND at BRIGADE, and increases the reorder quantities at all echelons (just as the ammunition NET LOSS activities do). The MEDEVAC activity moves person-

nel from ON HAND at BRIGADE to WAIT DSP at DISCOM, and increases the reorder quantity only at BRIGADE.

```
PIPE    BRIGADE   WPN CREW  NET LOSS  0.0     0.0     0.0     0.0
  FROM  BRIGADE   WPN CREW  ON HAND   1.0             1.0
  FROM  DISCOM    WPN CREW  REORDER   -1.0            -1.0
  FROM  COSCOM    WPN CREW  REORDER   -1.0            -1.0
  FROM  BRIGADE   WPN CREW  REORDER   -1.0            -1.0
PIPE    DISCOM    WPN CREW  MEDEVAC   6.0     0.0     6.0     0.0
  FROM  BRIGADE   WPN CREW  ON HAND   1.0             1.0
  TO    DISCOM    WPN CREW  WAIT DSP  1.0             1.0
  FROM  BRIGADE   WPN CREW  REORDER   -1.0            -1.0
```

The IN HOSP and MEDEVAC Activities at Higher Echelons. Patients arriving at the WAIT DSP node at DISCOM will either enter the hospital at DISCOM, or be evacuated to COSCOM. Treatment in the DISCOM hospital is represented by the IN HOSP activity. This activity moves patients from WAIT DSP to ON HAND. Note in the example below that the coefficient in the TO record is 0.17, not 1.0. This means that only 17 percent of all patients entering the hospital return to ON HAND status.[6]

```
PIPE    DISCOM    WPN CREW  IN HOSP   185.0   1.11    185.0   1.11
  FROM  DISCOM    WPN CREW  WAIT DSP  1.0             1.0
  TO    DISCOM    WPN CREW  ON HAND   0.17            0.17
  COEF  DISCOM    MED HRS   AVAIL     24.             24.
```

The number of patients who can enter the DISCOM hospital is limited by the available MEDIC capacity. This capacity is represented

[6]This trick may be useful for other activities. For example, some percentage of supplies may be lost in transit, either to pilferage or to enemy action.

by the constraint resource DISCOM .. MED HRS .. AVAIL, which is initialized as follows:

```
CONSTR  DISCOM    MED HRS   AVAIL
  RHS   DISCOM    MEDIC     ON HAND   2.85            2.85
```

This capacity is shared among the IN HOSP activities for all personnel types, so the relative priorities of these activities will determine which types of personnel will be admitted to the local hospital.

If there is a shortage of MEDIC capacity at DISCOM, the overflow of patients will be evacuated to COSCOM by a MEDEVAC activity. MEDEVAC activities are not constrained in the test case, so all patients that cannot be treated will be evacuated to the next higher echelon. These activities (one for each type of personnel) move patients from WAIT DSP at DISCOM to WAIT DSP at COSCOM and increase the reorder quantity at DISCOM.

```
PIPE    COSCOM    WPN CREW  MEDEVAC   12.0    6.11    12.0    6.11
  FROM  DISCOM    WPN CREW  WAIT DSP  1.0             1.0
  TO    COSCOM    WPN CREW  WAIT DSP  1.0             1.0
  FROM  DISCOM    WPN CREW  REORDER   -1.0            -1.0
```

Similar IN HOSP activities occur at COSCOM and THEATER, and a similar MEDEVAC activity moves overflow patients from COSCOM to THEATER. Examples of these activities are not shown here.

The RESUPPLY Activities. RESUPPLY activities transport personnel from rearward echelons to forward ones to fill requisitions.

For example, the COSCOM resupply activity moves a person of some type from ON HAND at THEATER to ON HAND at COSCOM.

```
PIPE     COSCOM     WPN CREW   RESUPPLY   24.0     18.1     24.0     18.1
  FROM   THEATER    WPN CREW   ON HAND    1.0               1.0
  TO     COSCOM     WPN CREW   ON HAND    1.0               1.0
  FROM   COSCOM     WPN CREW   REORDER    1.0               1.0
  COEF   THEATER    SUP HRS    AVAIL      1.0               1.0
  COEF   THEATER    TRCK HRS   AVAIL      0.6               0.6
  COEF   THEATER    DRVR HRS   AVAIL      1.2               1.2
```

It also reduces the reorder quantity at COSCOM and consumes specified amounts of supply personnel capacity, truck capacity, and driver capacity. These capacities are dealt with in the same fashion as the ordnance personnel capacity, truck capacity, and driver capacity were dealt with for the ammunition resupply activities. In fact, the truck and driver capacities refer to the same constraint resources here as in the ammunition resupply case. That is, in the test case, personnel resupply and ammunition resupply must compete for the same trucks and drivers.

Similar resupply activities transport personnel from COSCOM to DISCOM, and from DISCOM to BRIGADE.

The Support Structure for Equipment Recovery and Evacuation

Figure 12 shows what may happen to an item of combat equipment that is hit in battle. The test case includes activities that implement this network for each type of combat equipment shown in Table 1. These activities are not replicated for trucks, the only kind of equip-

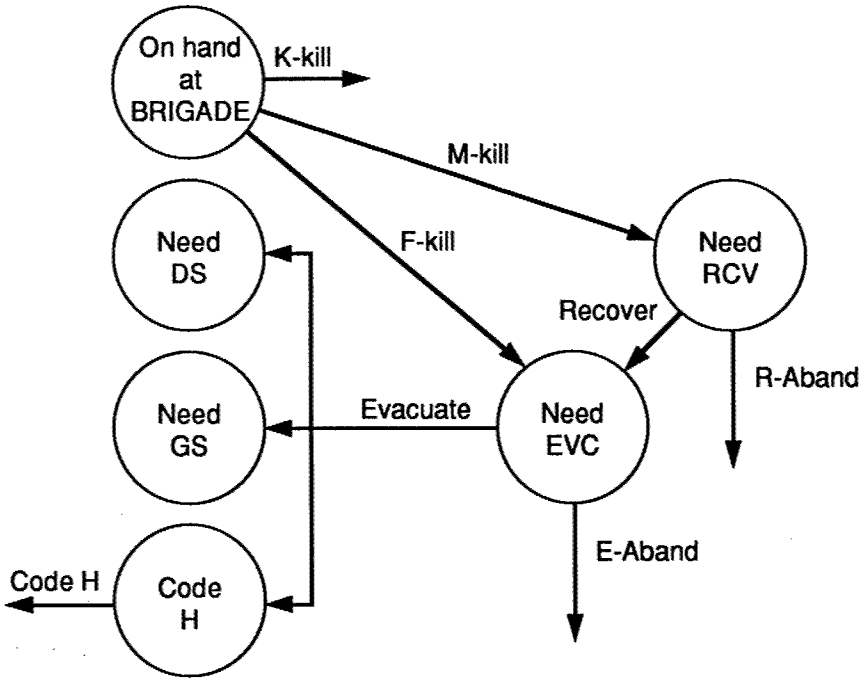

Fig. 12—Losses, Recovery, and Evacuation of Equipment

ment that the test case does not consider to be combat equipment. For tanks, the activities are:

```
BRIGADE   TANK      K-KILL
BRIGADE   TANK      M-KILL
BRIGADE   TANK      F-KILL
BRIGADE   TANK      RECOVER
BRIGADE   TANK      R-ABAND
BRIGADE   TANK      EVACUATE
BRIGADE   TANK      E-ABAND
BRIGADE   TANK      CODE H
```

The K-KILL, M-KILL, and F-KILL Activities. The K-Kill (catastrophic kill) activity is like the net loss activity for ammunition of personnel resources. For example, the K-KILL activity for tanks is:

```
PIPE    BRIGADE   TANK      K-KILL    0.0     0.0     0.0     0.0
  FROM  BRIGADE   TANK      ON HAND   1.0             1.0
  FROM  DISCOM    TANK      REORDER   -1.0            -1.0
  FROM  COSCOM    TANK      REORDER   -1.0            -1.0
  FROM  BRIGADE   TANK      REORDER   -1.0            -1.0
```

A catastrophic kill removes the weapon from the simulation and therefore reduces the number on hand of the corresponding equipment resource. It also increases by one the number of that equipment that the BRIGADE and all rearward echelons can requisition (the REORDER quantities). These REORDER quantities are defined and initialized in the same manner as the REORDER quantities for ammunition and personnel.

An M-Kill (mobility kill) reduces by one the on hand number of the corresponding equipment resource, but it does not remove it from the

simulation. Instead, that item of equipment is moved to a status called NEED RCV—need recovery by a recovery vehicle.

```
PIPE    BRIGADE   TANK      M-KILL    0.0     0.0     0.0     0.0
  FROM  BRIGADE   TANK      ON HAND   1.0             1.0
  TO    BRIGADE   TANK      NEED RCV  1.0             1.0
```

An F-Kill (firepower kill) also reduces by one the number of the equipment resource on hand. Unlike a mobility-killed vehicle, a firepower-killed vehicle can recover itself and hence does not need the services of a recovery vehicle. It makes its way unaided to a maintenance collection point (represented by the NEED EVC node in Fig. 12), from which it must later be evacuated with the aid of a HET. The F-KILL activity below therefore moves one vehicle (in this case a tank) from ON HAND status to NEED EVC status.

```
PIPE    BRIGADE   TANK      F-KILL    0.0     0.0     0.0     0.0
  FROM  BRIGADE   TANK      ON HAND   1.0             1.0
  TO    BRIGADE   TANK      NEED EVC  1.0             1.0
```

Each of these activities has a priority of zero and will be calculated in step 4 of the simulation cycle using CBT_LOSS data from the ATTRITION file.

The RECOVER and R-ABAND Activities. Before LDM recovers any M-KILLed vehicles, it must calculate the capacity to perform recovery operations. This capacity is represented by the constraint resource BRIGADE .. RCVY CAP .. RHS, which is initialized to zero at the start of every time period (note that there are no RHS records associated with the CONSTR record). However, the amount of this resource is increased from zero by the activity BRIGADE .. RCVY HRS .. AVAIL. Because this activity has a zero priority, its rate is calculated at step 4 from the CBT_LOSS data found in the ATTRITION file. If the reader will locate the DEP record with the name BRIGADE .. RCVY HRS .. AVAIL, he will find that this activity rate equals 12 times the engaged number of recovery vehicles, less six times the number of recovery vehicles hit. That is, each engaged

recovery vehicle supplies 12 hours of service, unless it is hit, in which case it supplies six fewer hours.

```
CONSTR  BRIGADE   RCVY CAP  RHS
PIPE    BRIGADE   RCVY HRS  AVAIL     0.0      0.0      0.0      0.0
  COEF  BRIGADE   RCVY CAP  RHS       -1.0               -1.0
```

The RECOVER activities (one for each type of combat equipment) move equipment from NEED RCV to NEED EVC status, and consume recovery capacity (the BRIGADE .. RCVY CAP .. RHS resource).

```
PIPE    BRIGADE   TANK      RECOVER   0.0      0.1      0.0      0.1
  FROM  BRIGADE   TANK      NEED RCV  1.0                1.0
  TO    BRIGADE   TANK      NEED EVC  1.0                1.0
  COEF  BRIGADE   RCVY CAP  RHS       1.0                1.0
```

Sometimes a vehicle cannot be recovered because there is too little recovery capacity available. Then the vehicle will be abandoned. The activity representing this for tanks in shown here. It reduces the number of tanks at the NEED RCV node and correspondingly increases the reorder quantities at all echelons.

```
PIPE    BRIGADE   TANK      R-ABAND   0.0      0.21     0.0      0.21
  FROM  BRIGADE   TANK      NEED RCV  1.0                1.0
  FROM  DISCOM    TANK      REORDER   -1.0               -1.0
  FROM  COSCOM    TANK      REORDER   -1.0               -1.0
  FROM  BRIGADE   TANK      REORDER   -1.0               -1.0
```

Note that the RECOVER activity has an earlier priority than the R-ABAND activity. In fact, the RECOVER activity for every equipment resource (see file BATTLE.DAT) has an earlier priority than the R-ABAND activity for any resource. This ensures that LDM will use up all recovery vehicle capacity before abandoning any combat vehicles in need of recovery.

The EVACUATE, E-ABAND, and CODE H Activities. Before LDM evacuates any vehicles, it must calculate the evacuation capacity. This is done in a manner entirely analogous to estimating the recovery capacity, except that the HET weapon is used in place of the RECOVERY weapon. That is, each engaged HET supplies 12 hours of service, unless it is hit, in which case it supplies six fewer hours.

```
CONSTR  BRIGADE   EVAC CAP  RHS
PIPE    BRIGADE   HET HRS   AVAIL     0.0      0.0      0.0      0.0
  COEF  BRIGADE   EVAC CAP  RHS      -1.0               -1.0
```

The activity representing evacuation of tanks is shown below. Similar activities for all the weapons can be found in the file BATTLE.DAT. The example activity moves a tank from NEED EVC status and consumes one hour of the HET weapon's time. The HET hours consumed must be established during calibration. Eighty percent of evacuated tanks are moved to the NEED DS node, 18 percent to the NEED GS node, and 2 percent to the CODE H node.[7]

```
PIPE    BRIGADE   TANK      EVACUATE  0.0      0.3      0.0      0.3
  FROM  BRIGADE   TANK      NEED EVC  1.0               1.0
  TO    BRIGADE   TANK      NEED DS   0.8               0.8
  TO    BRIGADE   TANK      NEED GS   0.18              0.18
  TO    BRIGADE   TANK      CODE H    0.02              0.02
  COEF  BRIGADE   EVAC CAP  RHS       1.0               1.0
```

Sometimes a vehicle cannot be evacuated because of a shortage of HETs. Then the vehicle will be abandoned. The activity representing this for tanks in shown here; it reduces the number of tanks at the

[7]Two levels of maintenance are represented in the test case. Direct support (DS) maintenance consists of fairly simple tasks, while general support (GS) maintenance involves more complex tasks requiring more maintenance resources per repair. Neither organizational nor depot maintenance is represented.

NEED EVC node and correspondingly increases the reorder quantities at all echelons.

```
PIPE    BRIGADE   TANK      E-ABAND   0.0     0.41    0.0     0.41
  FROM  BRIGADE   TANK      NEED EVC  1.0             1.0
  FROM  DISCOM    TANK      REORDER   -1.0            -1.0
  FROM  COSCOM    TANK      REORDER   -1.0            -1.0
  FROM  BRIGADE   TANK      REORDER   -1.0            -1.0
```

Vehicles that are sent to the CODE H node are unreparable. The CODE H activity serves to throw them away, and to increment the allowable requisitions at all echelons.

```
PIPE    BRIGADE   TANK      CODE H    0.0     0.51    0.0     0.51
  FROM  BRIGADE   TANK      CODE H    1.0             1.0
  FROM  DISCOM    TANK      REORDER   -1.0            -1.0
  FROM  COSCOM    TANK      REORDER   -1.0            -1.0
  FROM  BRIGADE   TANK      REORDER   -1.0            -1.0
```

The EVACUATE activities for every equipment resource has an earlier priority than the E-ABAND activity for any equipment resource. Thus, LDM will use up all evacuation capacity before abandoning any vehicle in need of evacuation.

The Support Structure for Equipment Repair and Resupply

Figure 13 shows the network of activities that represent the repair and resupply of combat equipment in the test case. The test case includes activities that replicate this network for each type of equipment in Table 1. These activities are:

```
DISCOM     TANK      GS EVAC
COSCOM     TANK      GS EVAC
THEATER    TANK      GS EVAC
DISCOM     TANK      DS EVAC
DISCOM     TANK      DS REPR
DISCOM     TANK      DS BKLOG
COSCOM     TANK      DS EVAC
COSCOM     TANK      DS REPR
COSCOM     TANK      DS BKLOG
THEATER    TANK      DS EVAC
THEATER    TANK      DS REPR
THEATER    TANK      DS BKLOG
THEATER    TANK      GS REPR
COSCOM     TANK      RESUPPLY
DISCOM     TANK      RESUPPLY
BRIGADE    TANK      RESUPPLY
```

All of these activities are needed for combat equipment types. For trucks, however, activities that involve the BRIGADE are not needed (DISCOM .. GS EVAC, DISCOM .. DS EVAC, and BRIGADE .. RESUPPLY). This network and the network shown in Fig. 12 join at the three nodes they have in common: ON HAND (at BRIGADE), NEED DS, and NEED GS.

The GS EVAC Activities. Equipment that arrives at the NEED GS node at the BRIGADE must be evacuated all the way to the THEATER to be repaired. (In the test case, GS maintenance capacity exists only at THEATER.) The equipment is evacuated in three stages: from BRIGADE to DISCOM, from DISCOM to COSCOM, and from COSCOM to THEATER.

The activity for evacuating tanks in need of GS repair from BRIGADE to DISCOM moves tanks from the NEED GS node at BRIGADE to the GS QUEUE node at DISCOM. It also increases the reorder quantity at BRIGADE by one for each tank evacuated.

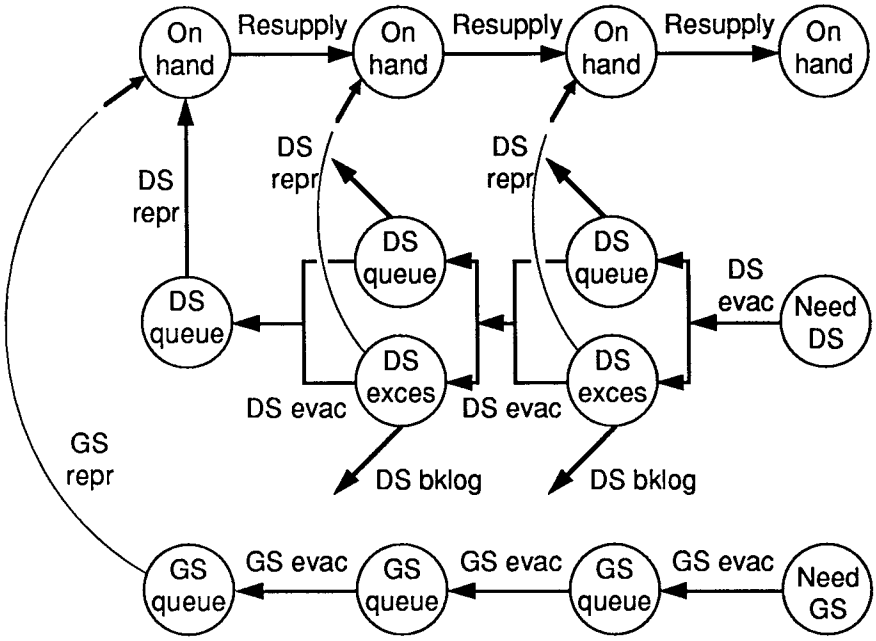

Fig. 13—Repair and Resupply of Combat Equipment

```
PIPE    DISCOM    TANK      GS EVAC    1.0      1.71     1.0      1.71
  FROM  BRIGADE   TANK      NEED GS    1.0               1.0
  TO    DISCOM    TANK      GS QUEUE   1.0               1.0
  FROM  BRIGADE   TANK      REORDER    -1.0              -1.0
```

The activity that evacuates tanks in need of GS repair from DISCOM to COSCOM moves tanks from GS QUEUE at DISCOM to GS QUEUE at COSCOM and increases the reorder quantity at DISCOM

by one for each tank evacuated. The activity also comsumes HET and driver capacity.

```
PIPE    COSCOM    TANK       GS EVAC   12.0    2.10    12.0    2.09
  FROM  DISCOM    TANK       GS QUEUE  1.0             1.0
  TO    COSCOM    TANK       GS QUEUE  1.0             1.0
  FROM  DISCOM    TANK       REORDER   -1.0            -1.0
  COEF  DISCOM    HET HRS    AVAIL     12.0            12.0
  COEF  DISCOM    DRVR HRS   AVAIL     24.0            24.0
```

Two constraint resources, one for HET capacity and one for driver capacity, are defined and initialized in the same way as shown earlier for ordnance personnel capacity. These resources are shared among the DS and GS evacuation activities for all types of combat equipment. (Driver capacity is also shared with ammunition and personnel resupply.) If HET or driver capacity is short at DISCOM, the relative priorities among the GS EVAC activities (and DS EVAC activities, which are discussed later) at this echelon will determine which types of equipment are evacuated during the current period and which types must wait until a later period for something to be done with them.

A similar activity evacuates tanks from COSCOM to THEATER. It differs from the evacuation from DISCOM to COSCOM only in that the THEATER has replaced the COSCOM, and the COSCOM has replaced the DISCOM.

The priorities of the GS evacuation activities dictate that tanks are evacuated from BRIGADE to DISCOM, then from DISCOM to COSCOM, and finally from COSCOM to THEATER. A tank that arrives at one echelon during a period can begin its journey to the next echelon in the same period. If the order of calculation were changed, the tank would have to wait for the next period at some point in the sequence.

Activities to Represent DS Repair at DISCOM. As shown in Fig. 13, each type of combat equipment has a DS QUEUE and a DS EXCES node at DISCOM. Initially, every item of equipment at the DS QUEUE is labeled excess. This is accomplished by defining a DS EXCES constraint resource for each type of equipment and setting its quantity equal to DS QUEUE in step 2 of the simulation cycle.

```
CONSTR  DISCOM    TANK      DS EXCES
  RHS   DISCOM    TANK      DS QUEUE  1.0              1.0
```

Equipment evacuated from the NEED DS node is added to both the DS QUEUE and DS EXCES at DISCOM. In the example below, this activity requires an average elapsed time of 1.0 hours.

```
PIPE    DISCOM    TANK      DS EVAC   1.0     4.61     1.0     4.61
  FROM  BRIGADE   TANK      NEED DS   1.0              1.0
  TO    DISCOM    TANK      DS QUEUE  1.0              1.0
  TO    DISCOM    TANK      DS EXCES  1.0              1.0
  FROM  BRIGADE   TANK      REORDER   -1.0             -1.0
```

Then a sequence of three activities is provided to process these tanks. The first activity is repair (the link DS REPR at DISCOM in Fig. 13). The rate of this activity is constrained by the availability of maintenance manhours (the MMH REPAIR constraint resource), which is initialized in step 2.

```
CONSTR  DISCOM    MMH       REPAIR
  RHS   DISCOM    MNT PSNL  ON HAND   2.85             2.85
```

The repair activity itself takes equipment out of both DS QUEUE and DS EXCES and moves it to the ON HAND node at DISCOM. In the process, it consumes a specified number of maintenance manhours (40 in the example below). Also, the activity requires a certain aver-

age elapsed time (10 in the example). Both the elapsed time and the maintenance manhours must be established during calibration.

```
PIPE    DISCOM    TANK      DS REPR   10.0    5.01    10.0    5.02
  FROM  DISCOM    TANK      DS QUEUE  1.0             1.0
  FROM  DISCOM    TANK      DS EXCES  1.0             1.0
  TO    DISCOM    TANK      ON HAND   1.0             1.0
  COEF  DISCOM    MMH       REPAIR    40.0            40.0
```

The second activity, labeled DS BKLOG in Fig. 13, takes tanks out of DS EXCES but leaves them in DS QUEUE. In effect, this activity (and analogous ones for other types of equipment) builds the backlog up either to its allowable limit or until all equipment items in the queue are accounted for, whichever happens first. The allowable limit is established by the constraint resource DISCOM .. MAINT .. QUEUE. In the example below, it has no RHS record and hence is initialized to zero at step 2 of the simulation cycle. The user might wish to specify the limit as a fixed number of manhours worth of backlog per maintenance person on hand; to do so, he would insert a record reading RHS .. DISCOM .. MNT PSNL .. ON HAND, and containing the desired numerical data.

```
CONSTR  DISCOM    MAINT     QUEUE
PIPE    DISCOM    TANK      DS BKLOG  0.0     6.01    0.0     6.02
  FROM  DISCOM    TANK      DS EXCES  1.0             1.0
  COEF  DISCOM    MAINT     QUEUE     40.0            40.0
```

Finally, an evacuation activity (the link DS EVAC starting at DISCOM in Fig. 13) takes tanks out of both DS QUEUE and DS EXCES at DISCOM and delivers them to the DS QUEUE and DS EXCES at COSCOM. This activity can take equipment from DS QUEUE only up to the number that are still at DS EXCES, so it will leave the backlogged amount (calculated by the DS BKLOG activity) at the DS QUEUE node. It also increases the allowable requisitions at DISCOM. This activity takes an average elapsed time of 12 hours in the example below; this parameter must be established during calibration. It also requires specified numbers of HET HRS and DRVR

HRS, two constraint resources defined earlier for use in the GS EVAC activity from DISCOM to COSCOM.

```
PIPE    COSCOM    TANK      DS EVAC   12.0    7.10    12.0    7.09
  FROM  DISCOM    TANK      DS QUEUE  1.0             1.0
  FROM  DISCOM    TANK      DS EXCES  1.0             1.0
  TO    COSCOM    TANK      DS QUEUE  1.0             1.0
  TO    COSCOM    TANK      DS EXCES  1.0             1.0
  FROM  DISCOM    TANK      REORDER   -1.0            -1.0
  COEF  DISCOM    HET HRS   AVAIL     12.0            12.0
  COEF  DISCOM    DRVR HRS  AVAIL     24.0            24.0
```

These activities are repeated at the COSCOM. The only changes are: THEATER replaces COSCOM and COSCOM replaces DISCOM; and at THEATER, there is no DS EXCESS node, since there is no choice to be made between holding equipment in a maintenance backlog and evacuating it to a rearward echelon.

DS and GS Maintenance at THEATER. DS repair at THEATER is simpler than DS repair at the forward echelons because there is no DS EXCES node. The activity moves equipment from the DS QUEUE to ON HAND status and consumes 40 maintenance manhours.

```
PIPE    THEATER   TANK      DS REPR   10.0    11.01   10.0    11.02
  FROM  THEATER   TANK      DS QUEUE  1.0             1.0
  TO    THEATER   TANK      ON HAND   1.0             1.0
  COEF  THEATER   MMH       REPAIR    40.             40.
```

GS repair of tanks at the THEATER is virtually identical to DS repair. The only differences are that it moves a tank from the GS QUEUE (rather than the DS QUEUE) to ON HAND status and it consumes 60 maintenance manhours rather than 40.

```
PIPE    THEATER   TANK      GS REPR   15.0    12.01   15.0    12.02
  FROM  THEATER   TANK      GS QUEUE  1.0             1.0
  TO    THEATER   TANK      ON HAND   1.0             1.0
  COEF  THEATER   MMH       REPAIR    60.             60.
```

These manhours will be shared among both the GS and DS repair at THEATER of all equipment items. If maintenance personnel are in short supply at the THEATER, the relative priorities of the repair activities for the various types of equipment will determine which equipment types are repaired immediately and which must wait until a later period. The quantity of available manhours is initialized in the same way as shown earlier for ordnance personnel capacity.

The RESUPPLY Activities. When an echelon loses equipment, or evacuates it to a rearward echelon for repair, it is entitled to requisition replacements. The RESUPPLY activities transport the replacements from one echelon to an adjacent one. For example, the activity that supplies tanks to the COSCOM takes tanks from the THEATER and adds them to the COSCOM. It also decrements the reorder quantity at the COSCOM and consumes some supply personnel capacity (the SUP HRS .. AVAIL resource shown below). Supply personnel capacity is initialized in the same way as shown earlier for ordnance personnel capacity.

```
PIPE     COSCOM     TANK       RESUPPLY  24.0     32.1     24.0     32.1
  FROM   THEATER    TANK       ON HAND   1.0               1.0
  TO     COSCOM     TANK       ON HAND   1.0               1.0
  FROM   COSCOM     TANK       REORDER   1.0               1.0
  COEF   THEATER    SUP HRS    AVAIL     1.0               1.0
```

The activities to resupply resources to the DISCOM and BRIGADE are essentially the same as the above activity that resupplies the COSCOM.

STEP 6: WRITE OUTPUT QUANTITIES

Any LDM simulation will calculate far too many quantities to examine exhaustively. Thus, the user must specify (in OUT_SPEC files) which quantities he will look at. He will wish to look at:

- Quantities that measure combat performance, such as FLOT movement, attrition, and consumption and losses of resources; and
- Quantities that suggest what factors limit combat performance, such as equipment awaiting repair (suggests a shortage of maintenance), or ammunition awaiting transportation (suggests

a shortage of transportation capacity), or any resource that is exhausted by the end of a time period.

Four OUT_SPEC files are distributed with the test case. Two of them, CBTLIM.OUT and LOSSES.OUT, request output quantities that measure combat performance. The other two, BDERES.OUT and AMMO.OUT, request output quantities that suggest how combat performance might be improved. It will be helpful to have listings of these files at hand while reading this section.

The file CBTLIM.OUT specifies the COMBAT output option, described in Sec. 6. This option generates the number of each weapon (1) available for combat, (2) engaged in combat, and (3) hit during combat. The file LOSSES.OUT specifies the output quantities for each resource that are shown in Table 5. (FLOT movement is included in every output file.) Figure 14 shows some of these measures from the test case. Except for the engaged tanks, all the measures in the figure are cumulated over time. This is not done by LDM; rather, the results are imported into a commercial spreadsheet program (I use LOTUS 1-2-3) and cumulated there. Other calculations can be performed in the spreadsheet as well. For example, the temporary losses of tanks in the figure consist of the EVACUATE minus CODE H quantities, while the permanent losses consist of K-KILL plus R-ABAND plus E-ABAND plus CODE H.

The output quantities specified in the file BDERES.OUT should provide some initial clues for identifying what factors limit combat performance. Included are output quantities for each resource that can be found at BRIGADE, including amounts of resources authorized, amounts available for incorporation into weapons, and amounts

Table 5

Output Quantities Specified in LOSSES.OUT

Equipment	Personnel	Ammunition
K-KILL	NET LOSS	NET LOSS
M-KILL	MEDEVAC	
F-KILL	(from BRIGADE	
RECOVER	to DISCOM)	
R-ABAND		
EVACUATE		
E-ABAND		
CODE H		

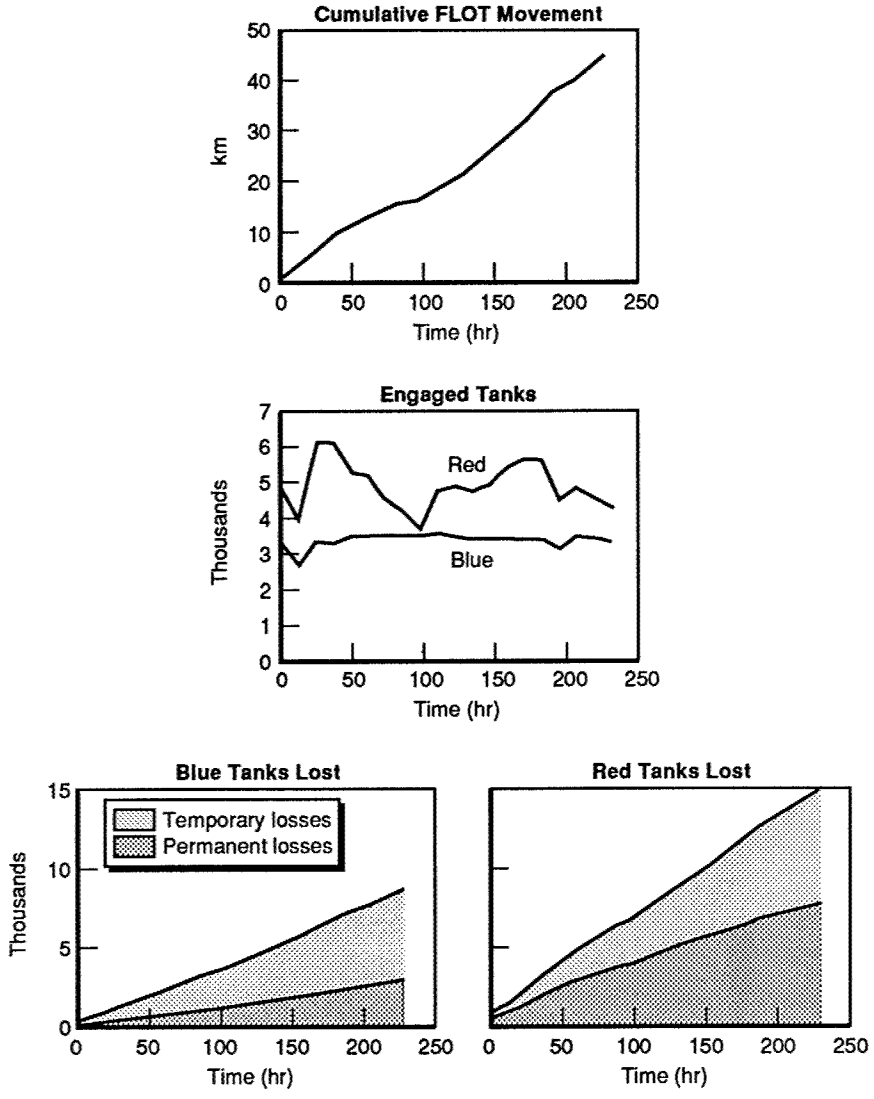

Fig. 14—Some Combat Performance Measures

remaining after as many weapons as possible have been formed from the resources. If one finds none of a given resource remaining after weapon formation, then adding more of this resource at BRIGADE might increase the number of weapons available for combat, hence improve combat performance. If the authorized and available amounts of such a resource are substantially the same, then the amount of this resource available at BRIGADE can be increased only if the authorized amount is first increased. This cannot be corrected by improvements in logistics. Thus, one should look for resources whose available amounts during a large part of the simulation are substantially less than their authorized amounts and whose amounts remaining after the weapon formation step are zero. In the test case, Blue has two such resources, tanks and artillery ammunition, as shown in Fig. 15.

The output quantities requested in the BDERES.OUT file do not reveal the reasons that these resources are in short supply at BRIGADE. To determine the reasons, outputs are needed that trace the behavior of these resources, and any other resources that may influence them, throughout the support structure. The file AMMO.OUT specifies output quantities that do this for artillery ammunition. I leave it to the reader to build an OUT_SPEC file that will trace the reasons for the shortage of tanks.

The AMMO.OUT file requests the authorized and available ammunition of each type at every echelon as well as the reorder quantities at all echelons but THEATER. An examination of these outputs shows that there is plenty of artillery ammunition available at all times and all echelons. Thus the shortage of artillery ammunition at BRIGADE is not due to a more widespread shortage. The AMMO.OUT file also requests the available capacity to handle ammunition at each echelon but BRIGADE, and the capacity remaining at the end of each period, when as much ammunition as desired or possible has been handled. At the THEATER and COSCOM echelons, there is ample ammunition handling capacity, but at the DISCOM, the capacity is exhausted in each period. The reason for the artillery ammunition shortage at BRIGADE is therefore a shortage of ammunition handlers at the DISCOM.

Although the test case is distributed with the four OUT_SPEC files discussed here, one would typically not know at the start of a study which output quantities ought to be requested. It is reasonable to make the initial LDM simulation run with OUT_SPEC files that provide measures of combat performance, like CBTLIM.OUT and

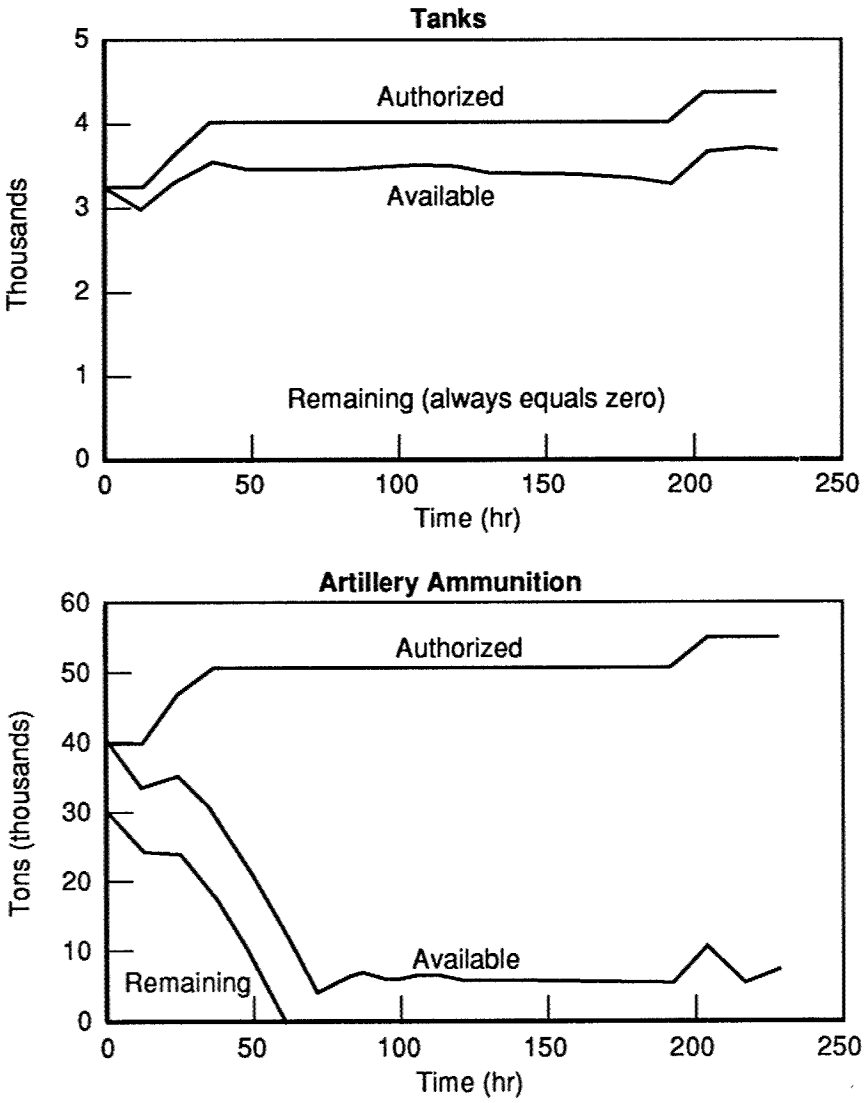

Fig. 15—Blue Resource Shortages at BRIGADE

LOSSES.OUT, plus an OUT_SPEC file that indicates which resources are in short supply at the BRIGADE echelon, like BDERES.OUT. However, at the time of the first simulation run, one will not know which resources are in short supply at BRIGADE and cannot therefore build OUT_SPEC files to trace the causes of those shortages. That must await a subsequent simulation run.

9. USING LDM FOR ANALYSIS

The POLA methodology, of which LDM is an important part, is intended to be used to analyze and compare possible logistics improvements. An analysis consists of the following steps, which are discussed in the following sections:

- Establish a base case that represents the current situation, or some other desired reference situation;
- Define excursion cases to estimate the effects on combat performance measures of possible logistics improvements to the base case;
- Design LDM cases to investigate the sensitivity of conclusions drawn from the excursion cases to changes in uncertain parameters and assumptions; and
- Estimate the cost of each logistics improvement, using other components of the POLA (or Logistics Net Assessment) methodology.

ESTABLISH A BASE CASE

It is important in any analysis to establish one case a *base case* to be a point of departure and comparison for all other cases. For an analysis with LDM, the base case should represent the Army (or an appropriate part of the Army) as of a certain time. For example, if one is analyzing logistics improvements to the U.S. Army in Europe as of 1989, the capabilities of the weapons, the support structure, and the inventories of resources in the base case should all reflect those of the U.S. Army in Europe as of 1989. The base case will ordinarily be constructed during the process of calibrating LDM (see Ref. [1], Secs. 2 and 3).

To specify any LDM case, one must specify its input files an ATTRITION file, one or more SUPPORT files, and a TIME_PHASE file. (It does not matter what OUT_SPEC files one specifies. They affect only the quantities that appear in the output files but do not influence the calculations done by LDM.) For the base case, these files should be as follows.

- The ATTRITION file should be the one developed during calibration to a CEM or FORCEM case for the appropriate year and

theater. If a completely appropriate ATTRITION file is not available, use one calibrated to a CEM or FORCEM (or other model) case from the same theater and a nearby year. Failing this, use what you must (but remember to take the analysis results as tentative).[1]

- The SUPPORT files must represent the support structure for that year and theater.
- The TIME_PHASE file should contain the resource inventories for the appropriate year and theater. It will typically be the one obtained from the calibration exercise. However, if one seeks to analyze the Army of a time later (or earlier) than the CEM or FORCEM calibration case, and in the interim some units have been reassigned or some new resources have become available, these changes can be incorporated in the TIME_PHASE file for the base case.

DEFINE EXCURSION CASES

What Logistics Modifications Can Be Analyzed?

LDM will be able to analyze a logistics modification only if there is a parameter in one of the input files that changes when the modification is made and if changing this parameter has an effect on some measure of combat performance calculated by LDM. Most logistics modifications involve increasing or decreasing resources or capacities, which would be represented by changes to the TIME_PHASE file. Other modifications involve changes in the durations of activities (e.g., transportation or repair times). These would be represented by changes in the SUPPORT files.

A logistics modification can be included directly or indirectly in the formulation of LDM's support structure. It is included directly if there is an explicit resource constraint in LDM that represents it. In the test case, for example, trucks were explicitly represented at each echelon, so one could represent the addition of trucks simply by incrementing the right-hand sides of the truck constraints.

[1]It would be desirable to be able to modify an existing ATTRITION file to reflect changes from one year or theater to another without having to completely recalibrate LDM. I suspect that the ATTRITION files for the same theater in two different years will be very similar, if the years are not far apart. I have no idea how the ATTRITION file will differ from one theater to another.

The modification is included indirectly if it is not included directly, but it nevertheless can be represented through its effect on a parameter in an LDM input file. For example, the LDM support structure may have a constraint representing the capacity of DS Ordnance companies to receive and issue ammunition at ammunition supply points (ASPs) (see Ref. [1], Sec. 5). Adding fork lifts to DS Ordnance companies should increase their capacity. Even though fork lifts are not explicitly mentioned as an LDM resource constraint, one could nevertheless represent their effect by appropriately adjusting the right-hand side of the DS Ordnance company capacity constraint.

Translating from Logistics Modifications to LDM

A preprocessor is often needed to translate a real-world logistics modification into terms the model can use. It may also be necessary to operate the preprocessor in reverse—to translate from the model's terms back to real-world terms. Reference [3] discusses some methods upon which such a preprocessor can be based.

Even for modifications directly represented in LDM, the preprocessor need not be straightforward. For example, suppose trucks are explicitly represented in LDM and that one of the real-world modifications to be analyzed is the addition of trucks. But the truck represented in LDM will be a generic truck (e.g., an equivalent ten-ton truck), while real-world trucks may have a variety of capacities. It will be necessary to calculate an equivalent number of generic LDM trucks, perhaps by estimating the total truck capacity to be added and dividing by the capacity per generic LDM truck.

For resources indirectly represented, the preprocessor must be even more sophisticated. For example, suppose one of the real-world modifications is adding variable reach fork lifts to DS Ordnance companies, and the LDM parameter it affects is the capacity to handle ammunition at the ammunition supply point (ASP). Then one must devise a means to estimate the effect of adding fork lifts on that capacity parameter.

The ORSA Support Team at the LEA has already built a preliminary version of such a preprocessor as part of their LNA system [4]. The LNA system consists of an input processor, the LDM program with the input files for a base case, an output analyzer, and a graph generator, all tied together into an integrated system by means of DOS batch files. The LNA input processor performs the function of the preprocessor discussed here.

Excursion Cases Proposed by Others

Some excursion cases will be proposed by people unconnected with the LDM analysis. New equipment items are developed regularly, and TRADOC schools incorporate them into new TOEs. For example, the palletized load system (PLS) has been made the basis of new units to transport ammunition. The variable reach fork lift is intended to replace the existing type of fork lift in GS Ordnance companies, yielding greater capacity with fewer people. These alternatives should be identified, and LDM excursion cases should be defined for them. The preprocessor for translating real-world resources to model terms will be used in this task.

Excursion Cases Generated Within the Analysis

Excursion cases can also be defined as a part of the analysis. The user can run the LDM base case and obtain outputs that pinpoint the resources or capacity limitations that limit combat performance. He may select any limiting constraint (there may be several) and add a resource that will relax it. He should increase this resource by degrees, running LDM again for each increase until further increases have no further substantial effect.

To construct these cases and to assess their combat performance, the user must request the appropriate outputs. Typical combat performance measures are FLOT movement, attrition rates, and remaining forces. I defined two OUT_SPEC files for the test case that request outputs of these kinds, namely, the files CBTLIM.OUT and LOSSES.OUT.

Other LDM outputs will pinpoint the constraints that prevent these combat performance measures from improving. The first step is to determine what resources at the BRIGADE echelon are in short supply. For example, in the test case, the remaining quantity of ARTY AMM is zero throughout much of the simulation, so during that time some Blue artillery is prevented from engaging. As discussed in Sec. 8, the OUT_SPEC file BDERES.OUT is designed to help pinpoint which brigade resources are in short supply.

Once the resource shortages at BRIGADE have been identified, the user can track down what factors are causing the shortages. This will require additional output requests and may require that the user rerun the LDM case with different OUT_SPEC files to generate them. The new outputs should display the amounts of the short resource(s)

at all echelons and in all conditions, as well as the rates of activities involving those resources at each echelon. Thus in the test case, the OUT_SPEC file AMMO.OUT contains a list of the amunition-related outputs one might request.

Looking at these outputs may reveal one of three kinds of limitations that prevent more of the resource from reaching the BRIGADE echelon:

- A stock limitation is indicated if there is little or none of this resource anywhere in the system. The only remedy is to increase the amount of stock available.
- A capacity limitation is indicated if there is stock in the system, but it is in queues somewhere. Generally, a buildup of stock at a node—a queue—results from inadequate capacity to issue or transport stock from that node. Depending on the node, the capacity could be a repair, handling or transportation capacity. To determine which, further runs of the same case may be necessary, with different quantities specified for output.
- A policy limitation is indicated if neither a stock nor a capacity limitation can be found. For example, authorized quantities may be set too low, or priorities for repair may be unwise. I have little to say concerning how the user should respond to policy limitations.

Once a limitation has been discovered, the user should define several LDM cases in which that limitation is relaxed by successively larger amounts. I pointed out in Sec. 8 that the test case has a shortage of artillery ammunition at BRIGADE that is due to a shortage of ammunition handling capacity at DISCOM. The resource that determines ammunition handling capacity at DISCOM is DISCOM .. ORD PSNL .. ON HAND, so I construct a sequence of excursion cases by adding successively larger amounts of this resource to the base case. (I also increase the amount of BRIGADE .. ORD PSNL .. AUTH.) I can also extend this sequence in the opposite direction, decreasing the amounts of these resources from the base case.

The coefficients in the SUPPORT file PER_SUP.DAT imply that one ORD PSNL could handle approximately 200 units of ammunition per 12-hour simulation cycle, or 20 tons per day. (A unit of ammunition is 100 lb.) It happens that in the test case, ten Blue divisions are present at time zero, increasing to 14 divisions by time 204 hours. At time zero, there are 1080 ORD PSNL at DISCOM (see Table 2), with a capacity to handle 21,600 tons of ammunition per day, or 2160 tons per day per division.

My excursion cases will extend upward and downward from this point in increments of 20 percent of the base case quantity, which is 360 ORD PSNL or 720 tons per day per division. This is only the increment I use at time zero. As more Blue divisions enter later in the simulation, more ORD PSNL also enter at DISCOM. Table 6 shows the cumulative numbers over time of ORD PSNL at DISCOM for each excursion case. If the user examines the file NOMTIME.DAT, the TIME_PHASE file for the test case, he will find the numbers of ORD PSNL at DISCOM to be the numbers in the column of Table 6 labeled "Base" are introduced at the times indicated. To create one of the excursion cases, the user need only replace those numbers, for both authorized and on hand ORD PSNL, with the numbers from one of the other columns.

Table 7 shows the ammunition handling capacity per division that these ORD PSNL provide. The "Number of Divisions" column shows the cumulative number of Blue divisions in the simulation at the indicated time. Each "Handling Capacity" column shows the cumulative ammunition handling capacity available at DISCOM for one of the cases, calculated as 20 tons per day for each ORD PSNL on hand, divided by the number of divisions. The divisions that arrive later do not bring with them as much capacity per division as is available at time zero.

Figure 16 shows some example results from these six cases. Ammunition consumption by Blue behaves as one might expect. For a short initial period, while the divisions draw down their basic loads, ammunition consumption is independent of ammunition handling capacity. But the less the handling capacity, the sooner and more severely is ammunition consumption curtailed.

Table 6

Number of ORD PSNL in the Excursion Cases

Time (hr)	Number of ORD PSNL Added to Case					
	E0	E1	E2	Base	E4	E5
0	0	360	720	1080	1440	1800
24	0	60	120	180	240	300
36	0	30	60	90	120	150
204	0	30	60	90	120	150

Table 7

Ammunition Handling Capacity per Division in the Excursion Cases

Time (hr)	Divisions	Handling Capacity (tons/division-day) in Case					
		E0	E1	E2	Base	E4	E5
0	10	0	720	1440	2160	2880	3600
24	12	0	700	1400	2100	2800	3500
36	13	0	692	1385	2077	2769	3462
204	14	0	686	1371	2057	2743	3429

FLOT movement and total (permanent plus temporary) Blue tank losses are unaffected by increases in handling capacity above the base case. An examination of the excursion cases would reveal that in these cases, only artillery ammunition consumption is curtailed. Thus, artillery is the only type of weapon whose engaged number is affected, and artillery contributes little combat worth to the total force. Decreases in handling capacity from the base case do affect FLOT movement and Blue tank losses, because in these cases, the consumption of both artillery and tank ammunition is curtailed.

Total Red tank losses are hardly affected by changes in Blue ammunition handling capacity. Blue curtails his ammunition consumption by engaging fewer weapons. Engagements occur at force ratios more favorable to Red, which tends to *decrease* Red attrition. The improved force ratio, however, induces Red to become more aggressive. A larger fraction of the engaged forces adopt a Red Attack Blue Defend posture, and smaller fractions adopt the Static or Blue Attack Red Defend postures. This shift in postures tends to *increase* Red attrition as Blue ammunition consumption declines. The two effects nearly cancel one another.

The results in Fig. 16 suggest that increases in ammunition handling capacity are of little value to Blue. However, decreases will cause him to lose ground to Red (cases E0, E1, and E2). At the same time, decreases in ammunition handling capacity cause Blue to withhold weapons from combat and therefore to suffer fewer losses. The user must decide whether loss of territory or the benefit of decreased losses is more important.

Several sequences of excursion cases can be generated by the process described above. At any point, there may be several constraints that

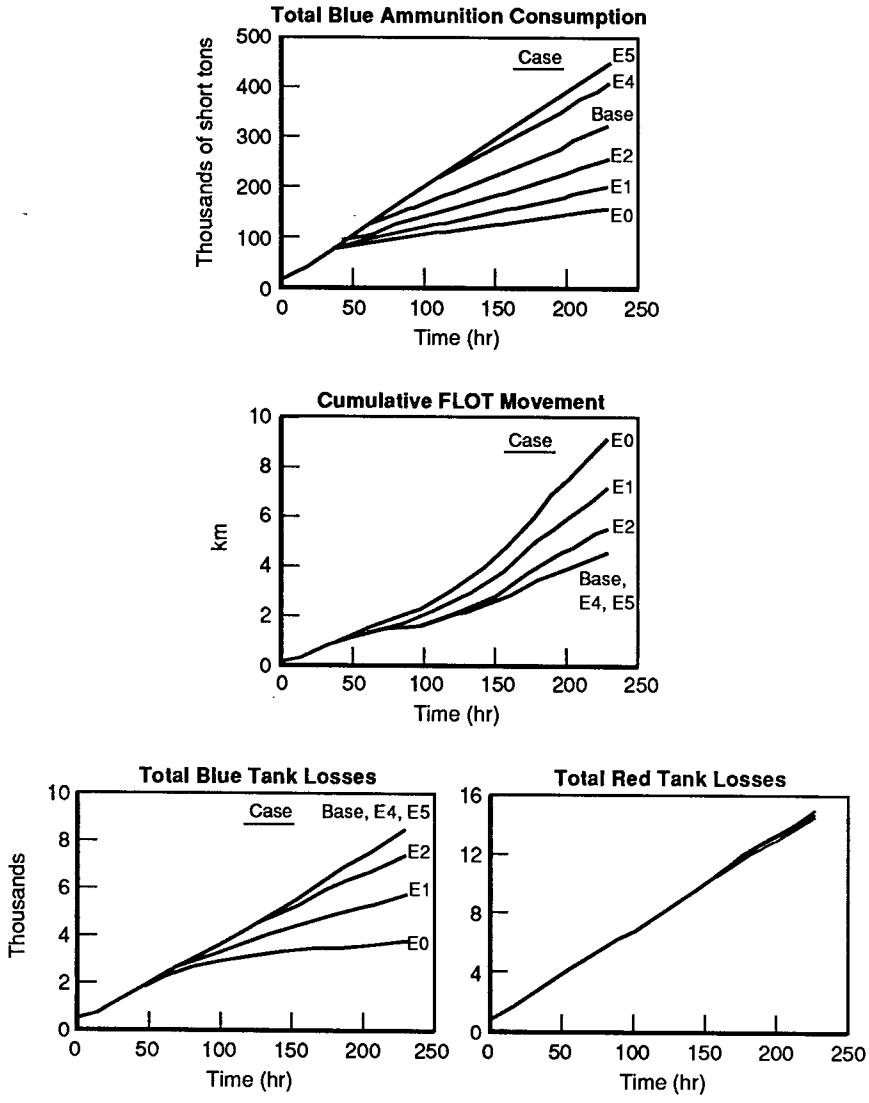

Fig. 16—Results from Excursion Cases

limit combat performance, and any of them may be chosen. In addition, there may be several resources whose addition will relax a given constraint. For example, Figure 15 shows that Blue has a shortage of tanks at BRIGADE that may be alleviated by an increase in war reserve stocks of tanks, or by an increase in repair capacity. The user need not choose one of these options over the other. He may instead choose to investigate both further.

It is not immediately evident what mix of real-world resources corresponds to an excursion case defined in this way. The LDM preprocessor mentioned above must be operated in reverse to generate a real-world resource mix that matches the excursion case. Some manual intervention may be needed to accomplish this.

DESIGN SENSITIVITY CASES

Sensitivity cases are intended to test the effects of changes in uncertain parameters on the benefits estimated for logistics improvements. If a particular logistics improvement is *robust*, it will make little difference how the uncertain parameters vary. That improvement will still yield substantial benefits. If the improvement is not robust, changing the uncertain parameters may eliminate the benefit. These parameters will typically be in the ATTRITION or SUPPORT files.

Example Changes in the ATTRITION File

The ATTRITION file contains CBT_LOSS coefficients that determine the consumption rate of each type of ammunition. There is a great deal of disagreement about the correct ammunition consumption rates, and the user might wish to try several rates in different LDM cases. Also in the ATTRITION file are the combat worth coefficients for each type of weapon. In the test case, the combat worth of artillery is very small, and yet artillery is responsible for by far the largest share of ammunition consumption. A user might wish to determine the effect on combat performance of increasing the combat worth of artillery relative to that of other weapons.

Example Changes in the SUPPORT Files

Many coefficients in the SUPPORT files represent the capacity of a resource to perform some function. For example, in the test case, each truck can provide 12 truck hours per 12-hour simulated period. One might wish to reduce this to allow time for routine maintenance. Similarly, transporting one unit of ammunition from COSCOM to DISCOM requires 0.06 truck hours in the test case. One might assume that 40 percent of a truck's time is wasted carrying things to what turns out to be the wrong place. Then that coefficient should be increased from 0.06 to 0.1 truck hours.

Other candidates for sensitivity cases could involve losses of support resources from enemy air attack. Each transportation activity might arrive at its destination with only 95 percent of the supplies it started with. This could be implemented by changing the coefficient in the appropriate TO records as follows:

```
PIPE    DISCOM    TANK AMM  RESUPPLY  12.0    15.3    12.0    15.3
  FROM  COSCOM    TANK AMM  ON HAND   1.0             1.0
  TO    DISCOM    TANK AMM  ON HAND   0.95            1.0
```

A fixed percentage of ammunition might be lost from each ON HAND node of the supply network. This would be implemented by:

```
STOCK   DISCOM    TANK AMM  ON HAND
  RHS   DISCOM    TANK AMM  ON HAND   -0.04           0.
STOCK   COSCOM    TANK AMM  ON HAND
  RHS   COSCOM    TANK AMM  ON HAND   -0.02           0.
STOCK   THEATER   TANK AMM  ON HAND
  RHS   THEATER   TANK AMM  ON HAND   -0.01           0.
```

In this example, Red loses no ammunition, and Blue loses 4 percent per 12-hour period of the ammunition at DISCOM, 2 percent at COSCOM, and 1 percent at THEATER.

Computing Sensitivities

Four LDM cases are involved in calculating the sensitivity of a logistics improvement to changes in an uncertain parameter. They are:

1. The base case, which I denote by BASE;
2. An excursion case, denoted by IMP, in which the logistics improvement has been added to the base case;
3. A modified base case, denoted by MOD_BASE, which is identical to the base case except that the uncertain parameter takes on a new value; and
4. A modified excursion case, denoted by MOD_IMP, which is identical to the original excursion case except that the uncertain parameter takes on the same new value as in MOD_BASE.

When the uncertain parameter takes on its base case value, the benefits offered by the logistics improvement are found by comparing case IMP with case BASE. Many quantities, including FLOT movement, engaged weapons, and losses of resources, can be compared at various times during the simulation. The change in any of these quantities is a benefit due to the logistics improvement, which may be written as:

$$BEN = IMP - BASE \tag{5}$$

Similarly, when the uncertain parameter takes on its new value, the benefits offered by the logistics improvement will be found by comparing case MOD_IMP with case MOD_BASE. As before, each benefit may be written as:

$$MOD\_BEN = MOD\_IMP - MOD\_BASE \tag{6}$$

To determine how sensitive the benefits from the logistics improvement are to changes in the uncertain parameter, one compares BEN with MOD_BEN. If they differ by a small amount or a small percentage, then the benefits are insensitive to the parameter.

ESTIMATE THE COST OF EACH LOGISTICS IMPROVEMENT

LDM does not estimate costs, but cost estimation is a necessary part of any analysis. No matter how large a benefit a logistics improvement may yield, it may still turn out to be worse than an alternative improvement that is less costly. The POLA methodology currently includes a rudimentary cost model, which is discussed in Ref. [3].

DISPLAYING RESULTS

Displays—the most common being graphs and tables—should generally compare results from several cases and not merely results from a single case. For example, to illustrate the effect of changing ammunition handling capacity, one might construct a graph with several curves showing cumulative FLOT movement over time, each curve from a case with a different handling capacity. Typically, one of the cases will be the base case and the others excursion cases. Figure 16 shows several such families of curves for the test case. Or one might construct a table with one column for each case and a row for each combat outcome one wanted to show—e.g., cumulative FLOT movement at Day D+10, weapon systems of various types on hand at D+10, total repairs by D+10, total tank-days of combat between D and D+10, etc.

One can show sensitivities in a similar way. Imagine a graph showing a benefit measure calculated using Eq. (5). One curve might show the improvement in FLOT movement over time due to an increase in ammunition handling capacity, under the assumption that the ATTRITION and SUPPORT files were those of the base case. Other curves might show improvement in FLOT movement over time for the same increase in ammunition handling capacity, but assuming the ATTRITION and SUPPORT files for different sensitivity cases. Or one could build a table. Every entry in the table would give information about the same logistics improvement. Each column would contain results for a different set of ATTRITION and SUPPORT files, and each row would contain a different combat outcome. The table entries would be improvements in the various combat outcomes as calculated using Equation (5).

CASE MANAGEMENT

It should be evident from the previous section that an analysis may involve many—perhaps hundreds—of LDM cases. Each case has several input files, some of which it will share with other cases. Each case may have several output files and contribute, along with other cases, to numerous displays. It is a major problem to keep track of all of this so that the analysis can be documented when the time comes to do so.

Perhaps the most valuable tool for case management is the log. There should be a separate log for each kind of object: ATTRITION files, SUPPORT files, TIME_PHASE files, OUT_SPEC files, output files, LOTUS spreadsheets, LDM cases, and displays. The user may wish

to distinguish among several different kinds of displays (e.g., tables vs. graphs) and keep separate logs for each. He may also wish to distinguish among different kinds of SUPPORT files. In the test case, for example, there are three SUPPORT files (BATTLE.DAT, MAINT.DAT, and PER_SUP.DAT), each defining a different part of the support structure.

Each entry in a log records information about a single object. Typically, the log should contain:

- The NAME of the object. If it is a disk file, the name should be the file name. It may be wise to have a longer title as well;
- The DATE the object was created. Although it is a dangerous practice, sometimes a file will be corrected and saved on top of the old version. Then the date should be when the corrected version was first saved and not when the old version was created;
- The name of the PREDECESSOR. The user often creates an object by making changes to another object of the same type. Keeping track of predecessors allows one to reconstruct the history of the analysis. If the user is replacing a file with a corrected version, the predecessor's name and this file's name are identical;
- A brief description of the modification(s) made to the predecessor. If there is no predecesor (if this is the ultimate ancestor of all other objects of this type), omit this description; and
- Links to other types of objects. For example, a CASE will be linked to its ATTRITION, SUPPORT, and TIME_PHASE files. An output file will be linked to the CASE and the OUT_SPEC file that generated it. A LOTUS spreadsheet will be linked to the output file(s) from which it takes data.

There are many uses for these logs, but one arises when the user corrects an LDM input file—for example, the ATTRITION file. This will, of course, render invalid any case that has used the old ATTRITION file, but it will be easy to find these cases in the case log. It will also be easy to define equivalent cases, the same as the invalidated cases except that the new ATTRITION file replaced the old.

It is also essential to save everything. It is wonderful to observe how many data files may be produced in the course of an analysis. As they begin to fill up one's available disk space, and as one begins to run out of new names for data files, the temptation is all but irresistible to erase old files. I recommend that you archive the files instead and record in the log where you have archived them. The ability to regenerate an old case can prove to be very important.

REFERENCES

1. J. H. Bigelow, T. Martin, and R. L. Petruschell, *Performance-Oriented Logistics Assessment (POLA): Preparing the Logistics Decision Model (LDM) for Use in Analyses*, RAND, N-3393-A, 1991.
2. J. H. Bigelow, *Performance-Oriented Logistics Assessment (POLA): Executive Summary*, RAND, R-3823-A, 1991.
3. J. H. Bigelow, T. Martin, and R. L. Petruschell, *Performance-Oriented Logistics Assessment (POLA): Relating Logistics Functional Capacities to Resources and Costs*, RAND, N-3824-A, 1991.
4. "Logistics Net Assessment System: System User's Guide," ORSA Support Team, U.S. Army Logistics Evaluation Agency, New Cumberland, PA 17070, 21 July 1989.